THE UNIVERSITY
of EDINBURGH

Direct methods for deductive verification of temporal properties in continuous dynamical systems

Andrew Sogokon

Doctor of Philosophy

Laboratory for Foundations of Computer Science
School of Informatics
University of Edinburgh

2015

Abstract

This thesis is concerned with the problem of formal verification of correctness specifications for continuous and hybrid dynamical systems. Our main focus will be on developing and automating general proof principles for temporal properties of systems described by non-linear ordinary differential equations (ODEs) under evolution constraints. The proof methods we consider will work directly with the differential equations and will not rely on the explicit knowledge of solutions, which are in practice rarely available. Our ultimate goal is to increase the scope of formal deductive verification tools for hybrid system designs. We give a comprehensive survey and comparison of available methods for checking set invariance in continuous systems, which provides a foundation for safety verification using inductive invariants. Building on this, we present a technique for constructing discrete abstractions of continuous systems in which spurious transitions between discrete states are entirely eliminated, thereby extending previous work. We develop a method for automatically generating inductive invariants for continuous systems by efficiently extracting reachable sets from their discrete abstractions. To reason about liveness properties in ODEs, we introduce a new proof principle that extends and generalizes methods that have been reported previously and is highly amenable to use as a rule of inference in a deductive verification calculus for hybrid systems. We will conclude with a summary of our contributions and directions for future work.

Lay Summary

As engineers design and build ever-more complex systems to scale with the demands of the modern world, finding errors in their designs becomes increasingly challenging. Many of the systems, such as those used in aerospace, automotive and nuclear power industries, are safety-critical, meaning that incorrect operation due to design flaws in these systems may result in severe consequences, such as loss of life, injury, or radioactive contamination. Ensuring that safety-critical systems operate correctly is a very difficult task that has been the focus of scientific research for a number of decades.

In order to model the behaviour of systems in the physical world (such as the motion of an aeroplane, or the processes taking place inside a nuclear reactor) engineers often employ the mathematical framework of ordinary differential equations. The goal of our thesis is to develop ways of answering questions about the behaviour of these mathematical models. For instance, if one has a mathematical model for a motion of two aeroplanes, it is desirable to know in advance whether a collision is possible; likewise, it may be helpful to know whether a nuclear reaction will lead to a reactor meltdown before it occurs, giving enough time to take preventive steps.

In this thesis we consider ways of demonstrating the properties of safety and eventuality in systems modelled by differential equations. By demonstrating safety one shows that it is impossible for an undesirable event to occur in the future, whereas with eventuality one shows that an event is inevitable at some point in the future. We also develop ways of automating the process of demonstrating safety properties and give an extensive survey of previous and related work.

Acknowledgements

During my time as a student at the LFCS, I have had the fortune and privilege of being advised by Dr. Paul B. Jackson, who I would firstly like to thank for his boundless patience, which doubtless underwent stress tests during the past four years. His politeness, encouragement and academic integrity have done much to support me throughout my time here and have made this thesis possible. I would also like to thank my second supervisor Dr. Jacques Fleuriot at CISA and Dr. Kousha Etessami at the LFCS for their judicious criticisms during our yearly review meetings, as well as my examiners Dr. Rob Arthan at Queen Mary, University of London and Dr. Ian Stark at the LFCS, for their careful reading and invaluable suggestions for improving this thesis.

I wish to extend my sincere gratitude to Sameed Zahoor Bhat and Marco Elver for ever being an inspiration for hard work, Ninna Stensgård at Linköping University for furnishing me with a rare copy of Krister Forsman's PhD dissertation, Jean-Luc Stevens and Hengjun Zhao for pointing me to some flaws in the manuscript, Mujahid Al-Adhami, Christian Buck, Gary Laidlaw and others for their time and company, Andreas Chatzistergiou for being a considerate office mate (and a terrific musician), as well as William Denman, Grant Passmore, James Bridge, Zongyan Huang, Dr. Eva Navarro-López and Prof. Lawrence C. Paulson for our discussions during visits to Cambridge and here in Edinburgh.

My warm thanks go to Jan-David and Christin Quesel for their kindness and hospitality and also to Dr. André Platzer for being a wonderful host during my brief visit to CMU, where I met with some brilliant people. I would especially like to thank Khalil Ghorbal, whose enthusiasm for the subject and care in explaining insights has kept my belief in the field. Thank you Khalil for our many technical discussions over Skype, without which I would have probably lost my marbles.

I would also very much like to thank my family, in particular my brother Paul for remaining close (and for our light-hearted discussions over a beer during my visits home) and finally give loving thanks to my mother Larissa, to whom I owe everything I have.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Andrew Sogokon*)

My mother's ambition to pursue a career in mathematics was not met with understanding or encouragement within her family and was subsequently unfulfilled. Throughout my life she worked very hard to ensure that I had not only the freedom, but the best opportunities to apply myself wherever I desired. Her support was hugely important to me throughout my studies at Edinburgh and during the writing of this document. Tragically, she did not quite live to see its final version complete. I dedicate this thesis to her memory.

Larissa Sogokon

1969 — 2016

Contents

Publications

Andrew Sogokon and Paul B. Jackson. Direct formal verification of liveness properties in continuous and hybrid dynamical systems. In *FM 2015: Formal Methods - 20th International Symposium, Oslo, Norway, June 24-26, 2015, Proceedings*, pages 514–531, 2015

Andrew Sogokon, Khalil Ghorbal, Paul B. Jackson, and André Platzer. A method for invariant generation for polynomial continuous systems. In Barbara Jobstmann and K. Rustan M. Leino, editors, *Verification, Model Checking, and Abstract Interpretation - 17th International Conference, VMCAI 2016, St. Petersburg, Florida, USA, January 17-19, 2016. Proceedings*, volume 9583 of *LNCS*. Springer, 2016

Khalil Ghorbal, Andrew Sogokon, and André Platzer. A hierarchy of proof rules for checking differential invariance of algebraic sets. In *Verification, Model Checking, and Abstract Interpretation - 16th International Conference, VMCAI 2015, Mumbai, India, January 12-14, 2015. Proceedings*, pages 431–448, 2015

Khalil Ghorbal, Andrew Sogokon, and André Platzer. Invariance of conjunctions of polynomial equalities for algebraic differential equations. In *Static Analysis - 21st International Symposium, SAS 2014, Munich, Germany, September 11-13, 2014. Proceedings*, pages 151–167, 2014

Paul B. Jackson, Andrew Sogokon, James P. Bridge, and Lawrence C. Paulson. Verifying hybrid systems involving transcendental functions. In *NASA Formal Methods - 6th International Symposium, NFM 2014, Houston, TX, USA, April 29 - May 1, 2014. Proceedings*, pages 188–202, 2014

Khalil Ghorbal, Andrew Sogokon, and André Platzer. A hierarchy of proof rules for checking positive invariance of algebraic and semi-algebraic sets. *Computer Languages, Systems & Structures*, 2015

Chapter 1

Introduction

1.1 Some motivating remarks

Formal deductive proofs are traditionally associated with efforts to formalise mathematics (such as Hilbert's program) put forward before Gödel's famous incompleteness results appeared in 1931. Though the idea of proving mathematical theorems by constructing explicit formal proofs using valid logical inferences from a set of axioms was historically dismissed by many (e.g. famously by Poincaré) as utterly impractical, the advent of powerful modern computers and improved software tools, such as interactive theorem provers, has meant that fully rigorous formal verification of non-trivial mathematical results is no longer an unattainable ideal. For instance, a formal proof of the Kepler Conjecture was sought by Hales et al. [HHM$^+$10] using HOL Light and was finally obtained in August 2014 (after some 11 years of continued effort). The Four Colour Conjecture and the Feit-Thompson Theorem were formally proved inside the Coq system by Gonthier [Gon08, GAA$^+$13]. The acceptance of formal proofs is still subject to fierce debates in the mathematical community; for instance, it may be questioned whether one can justify putting all trust in the correctness of a result into a formal proof that cannot realistically be checked by a human, or trust a machine to perform the check instead. There is, however, a general consensus that formal verification can, if anything, serve to provide greater assurances that a result is indeed correct, especially if a conventional proof already exists, but is very long and intricate.

Formal verification is not limited to checking results that are of purely mathematical interest. Modern control and computer engineering is (or at least ought to be) concerned with creating dependable systems whose behaviour is correct with respect to their desired specification. One need only recall such infamous examples of embarrassing and expensive failures as the 1994 Pentium floating-point bug [Pra95] or the failed 1996 Ariane Flight 501 to appreciate the argument for using more than the conventionally accepted paradigm of identifying design errors by extensive system testing.

Unlike proofs of certain results in mathematics, systems that one encounters in engineering are not commonly designed as objects of any intrinsic mathematical beauty that one would dread to obscure with tedious formalism. Furthermore, bugs in system designs are not uncommon and are sometimes even tolerated; their presence is seen as undesirable, but (depending on the nature of the bug) need not necessarily render the system altogether useless. One can see how formal verification, when viewed simply as a complementary stage in the design process that provides *greater assurances of correct operation*, or at least ensures absence of certain *critical bugs* in a system, has both great practical appeal and no potential to stoke up philosophic controversy regarding the nature of "greater assurance".

The use of formal verification on problems with industrial applications has largely focused on proving correctness properties of *discrete* systems (both hardware and software), where it has met with some success, e.g. in verifying floating-point division algorithms used in microprocessors [MLK98], cryptographic protocols [Pau98] and even large software projects, such as operating system kernels [KEH$^+$09] and compilers [Ler09]. *Continuous systems*, such as e.g. processes modelled by differential equations, have more recently become of significant interest to computer science and formal verification research, as they are often encountered in the context of a broader class of dynamical systems known as *hybrid* (or *cyber-physical*) systems. Research in hybrid systems is chiefly driven by the need for a unified approach to the design, simulation and verification of modern control systems. Today these can realistically involve the interaction of thousands of mechanical parts with electronic circuitry. A unified framework for modelling and analysing hybrid systems thus incorporates aspects of both computer and control engineering.

The behaviour of hybrid dynamical systems is typically determined by the interaction between the discrete controllers implementing *switching logic* and the physical environment, where state evolution is continuous. The class of systems that fall under this definition is incredibly broad (and naturally includes all systems that are purely discrete or purely continuous). To give but a few examples, hybrid systems have found application in modelling aircraft collision avoidance protocols [Pla10a, SAZ14], train control systems [Pla08, LLQ+10], control systems for oil drills working with discontinuous friction [NLC11], and many more. It is apparent that establishing correct operation of such systems is becoming increasingly important as well as increasingly difficult since, besides the challenges associated with verifying the correctness of discrete hardware and software, an extra layer of difficulty is added by also considering the continuous fragment, where the dynamics is typically governed by some (perhaps non-linear) system of differential equations. Indeed, verifying correctness specifications of hybrid systems is known to be a difficult task [FK04]; their expressiveness has been shown to make most interesting questions about their behaviour inherently undecidable [Hen96]. However, as with formalised mathematics, this does not mean that hybrid system verification is impossible and thus futile. On the contrary, formal verification tools have already been successfully applied in some impressive case studies (e.g. [ZYZ+14]), but it is also certainly true that there is great scope for improvement in what verification tools are currently capable of. In this thesis we will address some of the existing limitations in currently available formal verification approaches, with a focus on the continuous fragment.

1.2 Working hypothesis and goals of work

From a hybrid systems theory perspective, a *trivial* hybrid system is one which exhibits no hybrid behaviour whatsoever, i.e. does not combine discrete and continuous dynamics; a purely continuous system in which motion is everywhere described by a system of ordinary differential equations is an example (e.g. see [AHLP00, Example 5.1]). Yet, approaches to formally verifying hybrid systems often overlook the need for better verification tools for continuous systems. Instead, the focus has traditionally been placed on analysing hybrid systems in which the continuous dynamics is rather simple and any interesting

behaviour the system might produce originates from the interplay between its discrete and continuous parts. Such a state of affairs is particularly frustrating for potential users of verification technology who might attempt to apply currently available tools to prove rich temporal properties about hybrid systems in which continuous modes are governed by non-linear ordinary differential equations. Historically, the first systems to combine aspects of discrete and continuous control in a single modelling framework were *variable structure systems*, investigated by researchers in control theory since the 1950s [HGH93]. This class of hybrid dynamical systems has important applications in control engineering and often features relatively simple discrete switching logic and rich behaviour in the continuous fragment. Progress in verifying properties of such systems has been hampered by current limitations in handling the continuous fragment of hybrid systems. These stem from the nature of the solutions to ordinary differential equations, which are rarely available in closed-form. Even when closed-form solutions can be found, they frequently involve transcendental functions (such as $\sin, \cos, \exp, \ln$, etc.), which makes checking many important properties undecidable [Ric68]. The ordinary differential equations themselves will often possess a much simpler description than their respective solutions; it is thus natural to seek ways of answering questions about the temporal behaviour of continuous systems by working with the differential equations *directly*.

Direct methods for analysing properties of continuous systems began to emerge in the late nineteenth century. These developments were a by-product of the so-called *qualitative theory* of differential equations, which was pioneered by Poincaré, who (in a series of articles [Poi81, Poi82, Poi85]) initiated the study of methods for showing properties of differential equations without computing on their explicit solutions. Later work by Lyapunov [Lya92] used these ideas to study *stability* in dynamical systems and has ever since been associated with what are today known as Lyapunov functions, which are of central importance to modern control theory.

In essence, *Lyapunov's direct method* (also known as Lyapunov's second method [Par92]) gives a proof method that allows one to conclude that a trajectory (conventionally taken to start at the origin) of some given system of differential equations is *stable*. Informally, stability is a statement about the temporal behaviour of the system that requires all solutions initialised "near"

to the origin to remain "close" (see e.g. [BS70] or [HC08] for a comprehensive treatment). To apply the method, the user is required to find a certain (Lyapunov) function, conventionally denoted by V, that satisfies some formal criteria. If one finds an appropriate V and is able to check that it satisfies all the conditions prescribed by Lyapunov's second method, the stability property of the solution follows. Thus, in applying the method, one performs the following inference (from the *premise* above the bar to the *conclusion* below):

$$\frac{V \text{ is a Lyapunov function for the system}}{\text{The trajectory starting at the origin is stable}}$$

The power of the direct method lies in the fact that the formal conditions in the premise only involve statements about the function V and the differential equations, but *not* their solutions. Thus, provided that one can find a suitable function V, one has a formal criterion for stability that one can realistically hope to check. Practically speaking, this represents a *huge* improvement over trying to prove stability by working with the solution, which in the case of non-linear systems is almost never possible to even obtain as a finite expression. In the control community, Lyapunov's direct method is now considered a standard technique for demonstrating stability, though it took many decades after its original publication in 1892 for control engineers to fully appreciate its significance. The 1960s witnessed renewed interest in the method, which was stimulated by the difficult non-linear problems that began arising in control engineering at that time [Par92].

If we accept the fact that solving non-linear differential equations is not a realistic strategy for analysing their temporal behaviour, it is natural to ask: besides stability, what other temporal properties can we hope to verify *directly*, i.e. without ever having to solve the differential equations? In computer science, the two most commonly studied temporal properties of discrete systems are those of *safety* and *liveness*. Informally, a safety property states that "nothing bad happens", whereas a liveness property requires that "something good happens" during the operation of the system (see e.g. [AS87]). This motivates the following question for continuous systems, which will be the central theme of our thesis: can we formulate and mechanise general proof rules of the form

$$\frac{\text{Direct conditions for checking safety}}{\text{The system is safe}}, \quad \frac{\text{Direct conditions for checking liveness}}{\text{The system is live}} \ ?$$

Direct safety verification To verify safety of continuous systems, one can apply a technique familiar from program verification based on finding appropriate *inductive invariants* [TT09]. The main problem is, however, that of (directly) *checking* whether a given candidate defines an inductive invariant for a given continuous system. Many sufficient conditions for invariant checking have been proposed in the literature; we will give a very detailed survey and comparison along with our proposed extensions in Chapter 4 of this thesis. Recently, powerful methods have been developed by Liu, Zhan and Zhao [LZZ11] and Ghorbal [GP14a] that essentially solve the problem of direct invariant checking for a large class of continuous systems and invariant candidates. In order to mechanise formal proofs of safety that employ these methods, one still requires the means of *generating* suitable inductive invariants (much like one is required to come up with Lyapunov functions to prove stability using the direct method). In Chapter 5 we will develop algorithms for automatically *generating* inductive invariants for safety verification. Our approach is based on computing *exact* discrete abstractions of continuous systems, for which we extend previous work on discrete abstraction by Tiwari and Khanna [TK02, Tiw08a] by eliminating impossible transitions between discrete abstract states.

Direct liveness verification Liveness verification for continuous systems has generally received less attention in the literature than safety. Approaches to proving liveness properties in ODEs are often based on abstractions by timed automata [MB08, CNL12, SW11, SW13]. Although some direct verification approaches have previously been explored [PR05, Pla10a, SKA01, RS10], these are conservative in the sense that they often fail to prove liveness in a system where the property holds. In Chapter 6 of this thesis we will develop a very general direct proof method for verifying a type of liveness properties known as *eventuality* for non-linear continuous systems that extends and generalises methods available previously. We introduce the concept of *staging sets* and their associated *progress functions* to arrive at a rule of inference for eventuality verification with a decidable premise that is very well suited to use as part of a verification calculus for hybrid systems.

In the following chapter we will go over some mathematical preliminaries before reviewing continuous and hybrid dynamical systems in Chapter 3. In Chapter 7 we will summarise our contributions and discuss possible future directions.

Chapter 2

Mathematical preliminaries

Synopsis *Commutative algebra and algebraic geometry provide powerful computational tools for working with dynamical systems described using polynomials. In particular, recent results on invariant sets rely on established results from commutative algebra to give decision procedures for semi-algebraic invariant checking (which will become our concern in later chapters). In what follows, we will give a brief account of some important definitions, results and computational tools developed in algebraic geometry.*

2.1 Basic notions from commutative algebra

A *ring* is a mathematical structure that originally emerged as an abstraction of the set of integers $\mathbb{Z}$ with the usual operations of addition and multiplication.

Definition 1. *A commutative ring with identity is a set R closed under two binary operations $+ : R \times R \to R$ and $\cdot : R \times R \to R$ that satisfy the following ring axioms:*

$$\begin{aligned}
(a + b) + c &= a + (b + c) && \textit{associativity of } + \\
a + b &= b + a && \textit{commutativity of } + \\
\exists\ 0 \in R.\ \forall\ a \in R.\ a + 0 &= a && \textit{additive identity} \\
\forall\ a \in R.\ \exists\ -a \in R.\ a + -a &= 0 && \textit{additive inverse} \\
(a \cdot b) \cdot c &= a \cdot (b \cdot c) && \textit{associativity of } \cdot \\
a \cdot b &= b \cdot a && \textit{commutativity of } \cdot \\
\exists\ 1 \in R.\ \forall\ a \in R.\ a \cdot 1 &= a && \textit{multiplicative identity} \\
\forall\ a, b, c \in R. \quad a \cdot (b + c) &= a \cdot b + a \cdot c && \cdot \textit{ distributes over } +
\end{aligned}$$

Formally, such a structure is denoted by the triple $(R, +, \cdot)$.

Examples of rings include the integers under the usual operations, i.e. $(\mathbb{Z}, +, \cdot)$, the ring of complex numbers with complex addition and multiplication $(\mathbb{C}, +, \cdot)$ and univariate polynomials in the indeterminate x with coefficients in some ring R, i.e. the set of all formal expressions

$$a_0 \cdot x^0 + a_1 \cdot x^1 + a_2 \cdot x^2 + \cdots + a_k \cdot x^k,$$

where $k \in \mathbb{N}$ and $a_i \in R$ for all $0 \leq i \leq k$. In the latter example the two binary operations $+$ and $\cdot$ are taken to be *polynomial* addition and multiplication; by convention this ring is denoted $R[x]$. On the other hand, an example of a structure which is *not* a ring can be seen in the natural numbers $\mathbb{N}$ with the usual operations ($+$ and $\cdot$). This is because the natural numbers (except 0) do not possess an additive inverse that lies in $\mathbb{N}$, violating one of the ring axioms.

A less straightforward example of a commutative ring, but one which has many important practical applications, is the ring of *multivariate* polynomials in the

indeterminates $x_1, x_2, \ldots, x_n$ with coefficients in some ring R. The construction relies on the fact that starting with the coefficient ring R, we may build univariate polynomials in the first indeterminate x_1, forming the ring $R[x_1]$ that can be used as the *coefficient ring* for univariate polynomials in the second indeterminate x_2, i.e. $R[x_1][x_2]$. Repeating this process, we arrive at the ring $R[x_1][x_2]\ldots[x_n]$, which we write more concisely as $R[x_1, x_2, \ldots, x_n]$.

In practice, one often works with polynomials where the coefficient ring is in fact a *field*, such as the real numbers $\mathbb{R}$ or the rationals $\mathbb{Q}$. A field is a special kind of ring in which one can perform division. More formally, a field K is a commutative ring with identity which satisfies an additional axiom

$$\forall\ a \in K \setminus \{0\}.\ \exists\ a^{-1} \in K.\ a \cdot a^{-1} = 1 \qquad \text{multiplicative inverse.}$$

For instance, in the field of rational numbers $\frac{1}{2}$ is a multiplicative inverse of 2 since $\frac{1}{2} \cdot 2 = 1$. This axiom clearly does not hold for rings such as $\mathbb{Z}$ because $\frac{1}{2} \notin \mathbb{Z}$ and there is no non-zero $a \in \mathbb{Z}$ such that $a \cdot 2 = 1$. A field K is *algebraically closed* if any univariate polynomial in $K[x]$ has a root in K. For example, $\mathbb{C}$ is a typical example of an algebraically closed field because (by the fundamental theorem of algebra) any polynomial with complex coefficients has a complex root. This is not true of polynomials with real coefficients because they may possess no real roots, e.g. $x^2 + 1$ has real coefficients, but only complex roots.

An *ideal* is a subset of a ring that has important algebraic properties.

Definition 2. *Given a (commutative) ring R, a set $I \subseteq R$ is an **ideal** of R if and only if $\forall\ a, b \in I.\ a + b \in I$ and $\forall\ a \in I.\ \forall\ r \in R.\ r \cdot a \in I$.*

As an example of an ideal, consider the ring of integers $\mathbb{Z}$ and the set $E \subset \mathbb{Z}$ comprising all even integers. Clearly, the sum of two even numbers is again even, as is any integer multiple of an even number. Thus, E is an ideal of $\mathbb{Z}$. On the other hand, one sees that the set $O \subset \mathbb{Z}$ of odd integers does *not* form an ideal of $\mathbb{Z}$ because $3 \in O$ but $3 + 3 = 6 \notin O$.

A ring is called *Noetherian* if its ideals satisfy the *ascending chain condition*.

Definition 3 *(Ascending chain condition).* *Let R be a ring and let*

$$I_0 \subseteq I_1 \subseteq \cdots \subseteq I_N \subseteq \cdots$$

be any ascending sequence of ideals in R. Ideals in the ring R are said to satisfy the ascending chain condition if for any such sequence there exists an $N \in \mathbb{N}$ such that $I_{N'} = I_N$ holds for all $N' > N$.

Examples of Noetherian rings include $\mathbb{Z}$, $\mathbb{Q}$ and $\mathbb{R}$. The ring of continuous real-valued functions $C^1(\mathbb{R}, \mathbb{R})$ is an example of a ring which is *not* Noetherian.

Theorem 4 *(Hilbert's basis theorem).* *If R is a Noetherian ring, then the ring of univariate polynomials in the indeterminate x with coefficients in R, i.e. $R[x]$, is also Noetherian.*

Proof. See e.g. [Mis93, Proposition 2.3.6]. □

One may draw a simple corollary from Hilbert's basis theorem stating that the ring of multivariate polynomials $R[x_1, x_2, \ldots, x_n]$ is Noetherian if the coefficient ring R is Noetherian.

In general, an ideal may contain an infinite number of elements. However, Noetherian rings are special in that their ideals can always be represented using only a *finite set of generators*. An ideal $I \subseteq R$ is said to be *finitely generated* by $a_1, a_2, \ldots, a_k \in I$ if

$$\left(\sum_{i=1}^{i=k} r_i \cdot a_i \right) \in I$$

for any $r_1, r_2, \ldots, r_k \in R$, which is denoted by $I = \langle a_1, a_2, \ldots, a_k \rangle$. For instance, $\{2\}$ is the generating set for the ideal of even numbers in $\mathbb{Z}$, i.e. $E = \langle 2 \rangle$. In what follows, we will often work with multivariate polynomials with real coefficients, i.e. $\mathbb{R}[x_1, x_2, \ldots, x_n]$, which is a Noetherian ring whose ideals can always be represented using some finite set of polynomial generators $\langle p_1, p_2, \ldots, p_k \rangle$. In general, the generating set of a polynomial ideal need *not* be unique. However, Buchberger [Buc06] has shown that it is always possible to compute a special set of generators (known as the *Gröbner basis*; see [CLO10]) for ideals in polynomial rings, which leads to a *decision procedure* for checking whether a given polynomial $p \in \mathbb{R}[x_1, x_2, \ldots, x_n]$ is in some given ideal $\langle p_1, p_2, \ldots, p_k \rangle \subseteq \mathbb{R}[x_1, x_2, \ldots, x_n]$, i.e. one can decide whether or not $p \in \langle p_1, p_2, \ldots, p_k \rangle$.

2.2 Algebraic and semi-algebraic sets

As well as purely formal objects, polynomials in the ring $K[x_1, \ldots, x_n]$, can be viewed as functions with type $K^n \to K$. In this capacity, given some finite set of polynomials $I \subseteq K[x_1, \ldots, x_n]$, the set of all the common roots (in K^n) of polynomials in I, i.e.

$$\mathcal{V}(I) = \{\vec{x} \in K^n \mid p(\vec{x}) = 0,\ p \in I\},$$

defines an *algebraic set*, or an *affine variety*. In cases when $K = \mathbb{R}$, we write $\mathcal{V}_{\mathbb{R}}(I)$ and refer to this set as *real algebraic*, or a *real algebraic variety*. Note that if I is an ideal with infinitely many elements, the set of common roots of polynomials in I is precisely the set of common roots of the polynomials in the (finite) generating set $p_1, p_2, \ldots, p_k \in K[x_1, \ldots, x_n]$.

A more general class of subsets of $\mathbb{R}^n$, known as *semi-algebraic sets*, additionally admits polynomial *inequalities* in set descriptions. A semi-algebraic set S consists of a finite union of intersections of sets defined by polynomial equalities and inequalities, i.e. is of the form:

$$S = \{\vec{x} \in \mathbb{R}^n \mid \bigcup_{i=1}^{l} \bigcap_{j=1}^{m(i)} p_{ij}(\vec{x}) \sim_{ij} 0\}, \tag{2.1}$$

where $l \in \mathbb{N}$, $m : \mathbb{N} \to \mathbb{N}$, $p_{ij} \in \mathbb{R}[x_1, x_2, \ldots, x_n]$ and $\sim_{ij} \in \{=, <\}$.

Semi-algebraic sets may be represented and manipulated as purely formal entities, for which we will require some extra definitions.

A first-order *formal language* with equality consists of a finite set of constants *Consts* (e.g. $\{0, 1\}$), a set of relations *Rels* (e.g. $\{<\}$) and functions *Fns* (e.g. $\{+, -, \cdot\}$) defined for terms (*Trms*). Terms are constructed inductively as (i) elements of a countable set of variables *Vars* (e.g. $\{x_1, x_2, \ldots\}$) or constants *Consts*, (ii) $Fn(Trm_1, \ldots, Trm_k)$ where $Fn \in Fns$ is a function of arity k and $Trm_i \in Trms$ for each $i = 1, \ldots, k$. Atomic formulae *Atoms* are likewise defined inductively as (i) $Trm_1 = Trm_2$ or (ii) $Rel(Trm_1, \ldots, Trm_k)$, where $Rel \in Rels$ is a relation of arity k and $Trm_i \in Trms$ for each $i = 1, 2, \ldots, k$. From atomic formulae one can inductively construct first-order formulae *Forms*, which are defined as (i) atomic formulae in *Atoms*, (ii) $\neg\phi$, $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $\forall x.\ \phi$, $\exists x.\ \phi$, where $x \in Vars$ and $\phi, \phi_1, \phi_2 \in Forms$.

Remark 5. As a notational convenience to aid readability, one is free to define additional relation symbols $>, \geq, \leq$ in terms of $=$ and $<$, e.g. $Trm_1 > Trm_2$ is equivalent to $Trm_2 < Trm_1$, and the logical implication connective $\phi_1 \to \phi_2$ as syntactic sugar for $\neg\phi_1 \vee \phi_2$.

A variable that lies within the scope of a quantifier $\forall$ or $\exists$ is *bound* by that quantifier; otherwise, it is known as a *free variable*. For example, x is bound in the formula $\forall x.\ x > 0$, but is a free variable in $x^2 < 1$. A first-order formula with no free variables is known as a first-order *sentence*. For example, the following is a well-formed sentence in the first-order formal language $(<, +, -, \cdot, 0, 1)$ with equality:

$$\forall x_1.\ \exists x_2.\ x_1^2 = x_2 \wedge x_2 > 0.$$

A non-empty set providing an interpretation for the variables, together with interpretations for the formal functions *Fns*, relations *Rels* and constants *Consts* defines a *model* of the formal language. For instance, the set of real numbers $\mathbb{R}$ with the usual interpretations for the functions $+$, $\cdot$ and $-$, the order relation $<$ and the constants 0 and 1, which are taken to represent the additive and multiplicative identities (respectively), yields a model given by the structure

$$\mathfrak{R} = (\mathbb{R}, <, +, \cdot, -, 0, 1).$$

In a given model, any first-order sentence may either be true or false. The set of all true sentences in a model defines a *theory* for that model. In the case of $\mathfrak{R}$, the set of all true sentences defines the first-order theory of real arithmetic.

Tarski showed in [Tar51] that the first-order theory of real arithmetic is *decidable*, i.e. given any well-formed first-order sentence in the language of $\mathfrak{R}$, one can determine whether it is true or false using a terminating algorithmic procedure. Furthermore, Tarski has shown that $\mathfrak{R}$ admits *quantifier elimination*, i.e. given any well-formed formula ϕ (not necessarily a first-order sentence), there exists an equivalent *quantifier-free* first-order formula ψ, i.e. every variable occurring in ψ is a free variable and for any assignment of real values to the free variables ψ is true in $\mathfrak{R}$ if and only if ϕ is true (under the same free variable instantiation).

Any semi-algebraic set $S \subseteq \mathbb{R}^n$ may be characterized by some quantifier-free first-order formula ψ in the language of $\mathfrak{R}$, i.e.

$$S = \{(x_1, x_2, \ldots, x_n) \in \mathbb{R}^n \mid \psi(x_1, x_2, \ldots, x_n) \text{ is true } \},$$

(see e.g. Mishra [Mis93, Chapter 8]). For a general semi-algebraic set S given in (2.1), a natural choice for the quantifier-free formula ψ is:

$$\psi \equiv \bigvee_{i=1}^{l} \bigwedge_{j=1}^{m(i)} p_{ij} \sim_{ij} 0,$$

where, as in (2.1), $l \in \mathbb{N}$, $m : \mathbb{N} \to \mathbb{N}$, $p_{ij} \in \mathbb{R}[x_1, x_2, \ldots, x_n]$ and $\sim_{ij} \in \{=, <\}$.

To simplify our presentation, in this thesis we will interchangeably use the notation for sets and formulas characterizing those sets. Thus, S will denote both a semi-algebraic set $S \subseteq \mathbb{R}^n$ and a quantifier-free formula of real arithmetic with free variables in $x_1, \ldots, x_n$ that characterizes the set S.

2.3 Computational tools

Given a ring R, an element of this ring $r \in R$ and an ideal $I \subseteq R$, the ideal membership problem is to determine whether $r \in I$. Mayr and Meyer [MM82] have shown the ideal membership problem to be **EXPSPACE**-hard for the ring of multivariate polynomials with integer or rational coefficients. Mayr later proved the problem to be **EXPSPACE**-complete for polynomials over the rationals [May89]. The original algorithm proposed by Buchberger for constructing Gröbner bases, known today as *Buchberger's algorithm* (see e.g. [CLO10, Chapter 2, Section 7]), is implemented in most computer algebra systems and (inevitably) suffers from the high computational complexity associated with the problem. More recently, Faugère introduced Gröbner basis construction algorithms *F4* [Fau99] and *F5* [Fau02]. While theoretically these do not improve upon the complexity of Buchberger's algorithm in the general case, their practical performance is often observed to be very good [Fau02]. More recent work on for recent work on Gröbner basis algorithms may be found in [GGV10, PdMJ10].

Real quantifier elimination, that is the problem of producing an equivalent quantifier-free formula of real arithmetic, has been shown by Davenport and

Heintz [DH88] to exhibit doubly-exponential time complexity in the number of quantifier alternations. Existing algorithms for performing real quantifier elimination are often based in cylindrical algebraic decomposition (CAD), due to Collins [Col75], and its later improvements, such as *partial* CAD (PCAD) due to Collins and Hong [CH91]. The complexity of the CAD algorithm was shown by Collins [Col75] to be doubly-exponential in the number of variables. More details about CAD-based approaches to real quantifier elimination may be found in [CJ98]; an excellent introduction to CAD may also be found in [Jir98, Chapter 5]. Other approaches to performing real quantifier elimination, such as *virtual term substitution* (VTS), due to Weispfenning [Wei97, LW93], have been shown to be effective on classes of problems where the degrees of polynomials appearing in the formulas are suitably low (see [Bro05] for a good overview of VTS).

The class of purely universally-quantified or (equivalently) existentially-quantified sentences of real arithmetic, denoted by $\exists\mathbb{R}$, has been shown to exhibit singly-exponential time complexity in the number of problem variables [BPR96]. However, decision procedures for this class have so far proved to be less practical than the theoretically more complex decision procedures for the general problem based on CAD. More recent developments on decision procedures for $\exists\mathbb{R}$ may be found in [Pas11].

Techniques based on non-linear optimisation may also sometimes be used to determine the truth of universally-quantified sentences of real arithmetic. For instance, consider the following sentence:

$$\forall x_1.\ \forall x_2.\ \cdots\ \forall x_n.\ p(x_1, x_2, \ldots, x_n) < 0, \tag{2.2}$$

where $p \in \mathbb{R}[x_1, x_2, \ldots, x_n]$. If one can demonstrate that the global maximum of the function p is negative, i.e.

$$\max_{(x_1,x_2,\ldots,x_n)\in\mathbb{R}^n} p(x_1, x_2, \ldots, x_n) < 0,$$

then this would clearly imply that the universally-quantified sentence (2.2) is true. More involved sentences involving implication may also be handled this way by solving optimisation problems subject to constraints characterizing the premise (rather than all of $\mathbb{R}^n$, which is equivalent to a premise that is always true). In practice, the *convexity* of the function p (and the set of constraints) is often a crucial property that makes solving the optimization problem tractable;

non-convex optimization is much more difficult and is the focus of much ongoing research. Significant progress has been made in applying *sum-of-squares* and *semidefinite programming* methods to solve optimization problems of interest to engineering (see e.g. [Par00]).

Algorithms for solving quantified inequality constraints over the reals, based on *interval arithmetic*, have been developed and implemented by Ratschan [Rat06]. Later work by Gao et al. [GAC12] introduced so-called δ-complete procedures for existentially-quantified sentences. These approaches work by determining the *satisfiability* of quantified formulas of real arithmetic that do not feature any polynomial equalities (which may be encoded as non-strict inequalities). In order to apply these methods, one is required to bound the real variables inside real intervals, reducing the problem to searching for a satisfying variable assignment within some real hyper-box (rather than searching inside all of $\mathbb{R}^n$). For example, a typical problem is of the form:

$$\exists\, x_1 \in [0, 1].\ \exists\, x_2 \in [-50, 50].\ p_1(x_1, x_2) > 0 \wedge p_2(x_1) \leq 0 \vee p_3(x_1, x_2) \geq 0.$$

Notions of numeric stability and robustness are important to these approaches and one can quantify the maximum *perturbation* (δ) of the values assigned to the variables. We should note that these interval-based approaches are not restricted to purely polynomial problems and are able to work with formulas featuring transcendental functions. A competing approach, based on automated theorem proving and decision procedures for real arithmetic, has been developed by Akbarpour and Paulson [AP10] to handle the (generally undecidable [Ric68]) problem of determining the truth of universally-quantified sentences featuring transcendental functions.

Chapter 3

Continuous and hybrid dynamical systems

Synopsis *This chapter will give a brief overview of continuous, variable structure and hybrid dynamical systems. We review some basic notions and definitions, as well as specification formalisms. We conclude this chapter with a brief summary of existing hybrid system verification tools and approaches.*

3.1 Continuous systems

Differential equations often originate from practical applications, e.g. in mechanics, where they are used to describe the motion of physical artefacts; in biology, where they provide models for evolutionary processes; or can be completely abstract, of interest as purely mathematical models of deterministic finite-dimensional continuous processes.

A general ordinary differential equation (ODE) of order n is given by

$$F\left(t, x(t), \frac{dx}{dt}(t), \frac{d^2x}{dt^2}(t), \ldots, \frac{d^nx}{dt^n}(t)\right) = 0, \tag{3.1}$$

where t is the independent variable and $x(t)$ is an unknown n-times differentiable function of t. The *order* of equation 3.1 is defined to be the largest order of the derivative $\frac{d}{dt}$ appearing in the equation. An ordinary differential equation where the independent variable t does not appear explicitly is called *autonomous*.

Given an ordinary differential equation of order n (as in equation 3.1), it is always possible to transform it into a system of n *first-order* ordinary differential equations, following a method known as the d'Alembert transformation. By setting

$$x_1(t) = x(t),\ x_2(t) = \frac{dx}{dt}(t),\ \ldots,\ x_n(t) = \frac{d^{n-1}x}{dt^{n-1}}(t),$$

one can write down the following system of first-order ordinary differential equations:

$$\begin{aligned}
\frac{dx_1}{dt}(t) &= x_2(t),\\
\frac{dx_2}{dt}(t) &= x_3(t),\\
&\vdots\\
\frac{dx_n}{dt}(t) &= F(t, x_1, x_2, \ldots, x_n).
\end{aligned}$$

Remark 6. When working only with first derivatives, Newton's dot notation provides a more concise formalism than that of Leibniz; we shall henceforth simply write $\dot{x}$, rather than $\frac{dx}{dt}(t)$.

A general system of first-order ordinary differential equations is given by

$$\begin{aligned}\dot{x}_1 &= f_1(t, x_1, \ldots, x_n),\\ \dot{x}_2 &= f_2(t, x_1, \ldots, x_n),\\ &\vdots\\ \dot{x}_n &= f_n(t, x_1, \ldots, x_n).\end{aligned}$$

One can always eliminate explicit dependence on the independent variable in the system by setting $t = x_{n+1}(t)$ and introducing $\dot{x}_{n+1} = 1$ to obtain a larger *autonomous* system

$$\begin{aligned}\dot{x}_1 &= f_1(x_{n+1}, x_1, \ldots, x_n),\\ \dot{x}_2 &= f_2(x_{n+1}, x_1, \ldots, x_n),\\ &\vdots\\ \dot{x}_n &= f_n(x_{n+1}, x_1, \ldots, x_n),\\ \dot{x}_{n+1} &= 1.\end{aligned}$$

In this thesis, by a *system of ordinary differential equations* we shall mean an autonomous system with no explicit 'time'-dependence, given by

$$\begin{aligned}\dot{x}_1 &= f_1(x_1, \ldots, x_n),\\ \dot{x}_2 &= f_2(x_1, \ldots, x_n),\\ &\vdots\\ \dot{x}_n &= f_n(x_1, \ldots, x_n).\end{aligned}$$

We will denote such systems more concisely using vector notation as follows:

$$\dot{\vec{x}} = f(\vec{x}).$$

The main motivation for working with autonomous systems (besides their conceptual simplicity) is their geometric nature, which can often provide insights into their behaviour that would be obscured in the presence of independent variables. To develop this idea further, we will firstly need to introduce some terminology.

The set of states inside which a continuous dynamical system may evolve is known as the *phase space* (or *state space*) of the system. We will work with continuous

dynamical systems for which the phase space is given by the n-dimensional Euclidean space $\mathbb{R}^n$ and is thus a *differentiable manifold* [1]. Each state inside the state space is known as a *phase point*.

Given a phase space $\mathbb{R}^n$, an autonomous system of ordinary differential equations defines a *vector field* on $\mathbb{R}^n$ by assigning to each phase point $\vec{x} \in \mathbb{R}^n$ a vector $\vec{v}(\vec{x}) \in \mathbb{R}^n$, given by the right-hand side of the autonomous system evaluated at the phase point $\vec{x}$. That is, for each $\vec{x} \in \mathbb{R}^n$, the vector $\vec{v}(\vec{x})$ is defined by $\vec{v}(\vec{x}) = (v_1(\vec{x}), v_2(\vec{x}), \ldots, v_n(\vec{x}))$ where

$$\begin{aligned} v_1(x_1, x_2, \ldots, x_n) &= f_1(x_1, x_2, \ldots, x_n), \\ v_2(x_1, x_2, \ldots, x_n) &= f_2(x_1, x_2, \ldots, x_n), \\ &\vdots \\ v_n(x_1, x_2, \ldots, x_n) &= f_n(x_1, x_2, \ldots, x_n). \end{aligned}$$

We say that the system $\dot{\vec{x}} = f(\vec{x})$ induces the vector field $f : \mathbb{R}^n \to \mathbb{R}^n$ on the phase space and that for each phase point $\vec{x}$, the vector $f(\vec{x})$ gives the *phase velocity*.

When the size of the system (i.e. n) is small, one can visualise the vector field and observe the qualitative behaviour of the system. In doing this, one is typically less concerned with the vector field itself than the *direction* of the phase velocities.

A *Lie derivative* of a differentiable function $p : \mathbb{R}^n \to \mathbb{R}$ with respect to the system $\dot{x} = f(\vec{x})$ is the *directional derivative* of p in the direction of the vector field and is denoted $\mathfrak{L}_f(p)$. The Lie derivative is defined as

$$\mathfrak{L}_f(p) \equiv \sum_{i=1}^{n} \frac{\partial p}{\partial x_i} f_i(\vec{x}) \equiv \nabla p \cdot f(\vec{x}),$$

Higher-order Lie derivatives are defined inductively as $\mathfrak{L}_f^k(p) = \mathfrak{L}_f(\mathfrak{L}_f^{k-1}(p))$, with $\mathfrak{L}_f^0(p) = p$. Because each $f_i(\vec{x})$ gives the rate of change of x_i with respect to time, i.e. $f_i(\vec{x}) = \dot{x}_i = \frac{dx_i}{dt}$, we have

$$\mathfrak{L}_f(p) \equiv \sum_{i=1}^{n} \frac{\partial p}{\partial x_i} \frac{dx_i}{dt} \equiv \frac{dp}{dt}$$

[1] Intuitively, a differentiable manifold is a set endowed with a structure that allows one to do calculus. See e.g. [Chi06, KB00] for a more rigorous treatment.

i.e. the Lie derivative gives the time derivative of p. One may also define the Lie derivative as giving the rate of change of p if one were following the *solutions* to the system $\dot{\vec{x}} = f(\vec{x})$.

A *solution* to the initial value problem[2] $(\dot{\vec{x}} = f(\vec{x}),\ \vec{x}_0)$ is a function $\varphi : (a, b) \to \mathbb{R}^n$ such that $\varphi(t)|_{t=0} = \vec{x}_0$ and $\frac{d}{dt}\varphi(t)|_{t=\tau} = f(\varphi(\tau))$ for all τ in some non-empty extended real interval (a, b) including 0. We will denote solutions to the initial value problem at time $t \in (a, b)$ by $\varphi_t(\vec{x}_0)$, where $\vec{x}_0$ is the initial value. The interval $(a, b) \subseteq \mathbb{R} \cup \{-\infty, \infty\}$ is known as the *interval of existence* of a given solution; in what follows we will always consider the largest such interval, i.e. the *maximal* interval of existence.

Note that in general, solutions to initial value problems need not be unique or even exist for all time $t \geq 0$, i.e. the maximal interval of existence need not be of the form (a, ∞). For instance, solutions to simple non-linear systems, such as $\dot{x} = x^2$, already exhibit *finite time blow-up*, i.e. diverge to infinity in finite time.

For a given $\vec{x}_0 \in \mathbb{R}^n$, by mapping the maximal interval of existence (a, b) into the phase space $\mathbb{R}^n$ using the solution $\varphi_t(\vec{x}_0)$, one obtains a *phase curve* (also known as *orbit*) through the phase point $\vec{x}_0$. More formally, given a state $\vec{x}_0 \in \mathbb{R}^n$, the set

$$\gamma^+(\vec{x}_0) \equiv \{\vec{x} \mid \vec{x} = \varphi_t(\vec{x}_0),\ t \in [0, b)\}$$

is the positive semi-orbit through $\vec{x}_0$ under the flow mapping. Similarly, the set

$$\gamma^-(\vec{x}_0) \equiv \{\vec{x} \mid \vec{x} = \varphi_t(\vec{x}_0),\ t \in (a, 0]\}$$

is the negative semi-orbit through $\vec{x}_0$ under the flow mapping. The orbit through $\vec{x}_0$ is defined by

$$\gamma(\vec{x}_0) = \gamma^+(\vec{x}_0) \cup \gamma^-(\vec{x}_0).$$

The union of all the orbits in a system is known as the *phase portrait*.

Example 7 *Harmonic oscillator*
Simple harmonic motion can be observed in a mass m (kg) suspended from an elastic spring with stiffness constant k (as shown in Figure 3.1), which obeys Hooke's law, i.e. $F = -kx$, where F (N) is the force required to displace the mass by x (m) from the point of equlibrium.

[2] Also known as the Cauchy problem.

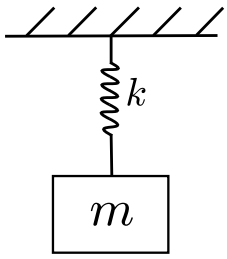


Figure 3.1: Harmonic oscillator.

From Newton's second law, $F = m\ddot{x}$, we obtain the following ordinary differential equation that models the motion of the mass on a spring:

$$m\ddot{x} + kx = 0,$$

which is more commonly rendered as

$$\ddot{x} + \omega^2 x = 0,$$

where $\omega = \sqrt{\frac{k}{m}}$ is the frequency of oscillation. By setting $x_1 = x$ and $x_2 = \dot{x}$ (d'Alembert), we may write this down as a system of first-order ODEs:

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= -\omega^2 x_1. \end{aligned}$$

Taking $\omega = 1$ and the initial condition $\vec{x}_0$ to be the phase point $(0, 1)$, the phase curve (orbit) through $\vec{x}_0$ corresponds to the unit circle centred at the origin (as shown in Figure 3.2).

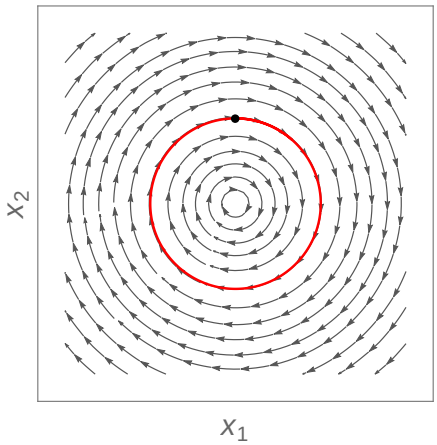


Figure 3.2: Phase curve (in red) through the phase point $(0, 1)$ (in black).

■

Given a system $\dot{\vec{x}} = f(\vec{x})$, the Picard-Lindelöf/Cauchy-Lipschitz theorem establishes the necessary and sufficient criteria for the existence of unique solutions to the initial value problem.

Definition 8. *A mapping $f : \mathbb{R}^n \to \mathbb{R}^n$ is* globally Lipschitz continuous *if there exists an $L \geq 0$ (known as the Lipschitz constant) such that for each $\vec{x} \in \mathbb{R}^n$ and for all $\vec{y} \in \mathbb{R}^n$ it is the case that*

$$\|f(\vec{x}) - f(\vec{y})\| \leq L\|\vec{x} - \vec{y}\|.$$

If for all $\vec{x} \in \mathbb{R}^n$ the mapping f is Lipschitz continuous in some neighbourhood U of $\vec{x}$, then f is said to be locally *Lipschitz continuous.*

Theorem 9 *(Existence and uniqueness theorem).* *If $f : \mathbb{R}^n \to \mathbb{R}^n$ is a locally Lipschitz-continuous mapping, then for every initial condition $\vec{x}_0 \in \mathbb{R}^n$ there exists a unique solution $\varphi_t(\vec{x}_0)$ to the initial value problem $(\dot{\vec{x}} = f(\vec{x}), \vec{x}_0)$, defined for all t in some non-empty interval (a, b), with $a < 0 < b$.*

Proof. The proof proceeds by defining an initial approximation to the solution and then recursively refining it in a process known as *Picard iteration*. See e.g. [Har64, Theorem 1.1]. □

Remark 10. Polynomial functions are locally Lipschitz continuous and as a consequence polynomial ODEs possess locally unique solutions.

In applications, one often encounters systems of ordinary differential equations under *evolution constraints*, i.e. of the form:

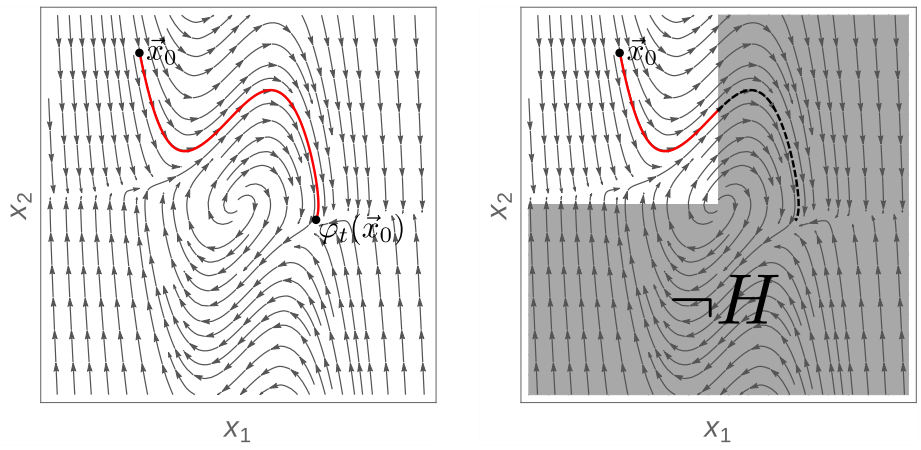
$$\begin{aligned}
\dot{x}_1 &= f_1(\vec{x}), \\
\dot{x}_2 &= f_2(\vec{x}), \\
&\vdots \\
\dot{x}_n &= f_n(\vec{x}), \\
\vec{x} &\in H \subseteq \mathbb{R}^n,
\end{aligned}$$

where the evolution domain constraint H is some (often semi-algebraic) set. We will write this concisely using vector notation as

$$\dot{\vec{x}} = f(\vec{x}) \;\&\; H.$$

Example 11 *Continuous system under evolution constraint*
Consider the Van der Pol system (with $\mu = 1$) in which evolution is confined to the fourth quadrant of $\mathbb{R}^2$, i.e.

$$\begin{aligned}\dot{x}_1 &= x_2,\\ \dot{x}_2 &= \left(1 - x_1^2\right) x_2 - x_1,\\ H &\equiv x_1 \leq 0 \wedge x_2 \geq 0.\end{aligned}$$



(a) Forward orbit segment in the Van der Pol system.

(b) Motion (in red) considered under the evolution constraint H (white).

Figure 3.3: Van der Pol system under evolution constraint H.

In this system, for a given initial condition $\vec{x}_0 \in \mathbb{R}^n$, any motion outside the evolution constraint H (e.g. Figure 3.3b, dashed) is not considered. In Figure 3.3 the state $\varphi_t(\vec{x}_0)$ is reachable from $\vec{x}_0$ after time t in the unconstrained system (Figure 3.3a), but is considered unreachable in the system under the evolution constraint H (Figure 3.3b). ■

In practice, evolution constraints are often used to define *operating modes* in hybrid and cyber-physical systems, which we will review in the next section.

3.2 Hybrid dynamical systems

Hybrid systems combine discrete and continuous behaviour and thus generalise both continuous and discrete dynamical systems. Many frameworks have been proposed to model such systems, with work in the control community often focusing on augmenting continuous systems to model discrete events [SP95, Fil88] and efforts in computer science largely directed at extending existing discrete computational models to also feature continuous state evolution [ACHH92, Pla10b].

Historically, the first attempt at modelling dynamical systems that fall under the definition of hybrid systems was made with the introduction of *variable structure systems*, given by differential equations with discontinuous right-hand sides [Fil88]. Variable structure systems (i.e. systems of the form shown in Figure 3.4) were introduced in the 1950s to model *switched* dynamical systems where mode switching is state-dependent [HGH93].

$$\dot{\vec{x}} = \begin{cases} f_1(\vec{x}) & \text{if } \vec{x} \in S_1 \\ f_2(\vec{x}) & \text{if } \vec{x} \in S_2 \\ \vdots & \\ f_n(\vec{x}) & \text{if } \vec{x} \in S_n \end{cases}$$

Figure 3.4: System with discontinuous right-hand side.

Many concepts of *solution* have been proposed for such systems [H$\acute{7}$9, SGT08], but intuitively one may think of piecewise-smooth functions which result in orbits that are obtained by simply concatenating the orbit segments spanning over different modes of the system (see e.g. di Bernardo et al. [dBBCK08]).

In applications, mode switching in variable structure systems occurs at the *boundary of discontinuity* given by some *switching surface* $s(\vec{x}) = 0$. A common variable structure control methodology is that of *sliding mode control* [Utk92],

which attempts to achieve system order reduction by steering trajectories onto a lower-dimensional space (sliding set) using a discontinuous control input.

$$\dot{x} = \begin{cases} f_+(\vec{x}) & \text{if } s(\vec{x}) > 0 \\ f_-(\vec{x}) & \text{if } s(\vec{x}) < 0 \end{cases}$$

Figure 3.5: Switched system

Example 12 *Switched system [HGH93]*
Consider the switching surface $s(\vec{x}) = 0$ defined by $x_1(\frac{x_1}{2} + x_2) = 0$ and let the system

$$\dot{x}_1 = x_2,$$
$$\dot{x}_2 = 2x_2 - 5x_1$$

be active in the region where $x_1(\frac{x_1}{2} + x_2) > 0$ is true and the system

$$\dot{x}_1 = x_2,$$
$$\dot{x}_2 = 2x_2 + 3x_1$$

govern the evolution inside the region where $x_1(\frac{x_1}{2} + x_2) < 0$ holds.

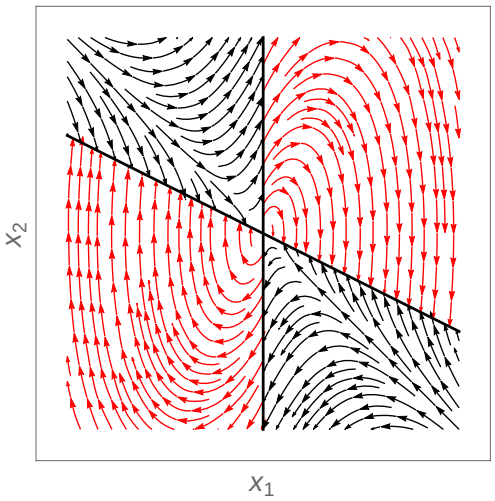


Figure 3.6: Phase portrait of a system with switching surface $x_1(\frac{x_1}{2} + x_2) = 0$.

■

Note that in systems such as that in Figure 3.5 the dynamics at the surface of discontinuity $s(\vec{x}) = 0$ is still undefined. Methods have been proposed for defining the dynamics at the surface of discontinuity in order to model *sliding behaviour* in the discontinuous system [Utk92]. This is typically achieved by constructing a new vector field $\dot{\vec{x}} = f_s(\vec{x})$, active on the boundary of discontinuity, in terms of $f_+(\vec{x})$ and $f_-(\vec{x})$ (see [UGS99, Fil88]), i.e. $\dot{\vec{x}} = f_s(\vec{x})$ if $s(\vec{x}) = 0$, where

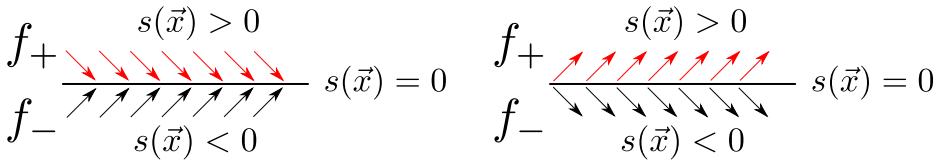
$$f_s(\vec{x}) = \frac{f_+(\vec{x}) + f_-(\vec{x})}{2} + u_{eq}(\vec{x})\frac{f_+(\vec{x}) + f_-(\vec{x})}{2},$$

$$u_{eq}(\vec{x}) = \frac{\nabla s(\vec{x}) \cdot f_+(\vec{x}) + \nabla s(\vec{x}) \cdot f_-(\vec{x})}{\nabla s(\vec{x}) \cdot f_-(\vec{x}) - \nabla s(\vec{x}) \cdot f_+(\vec{x})},$$

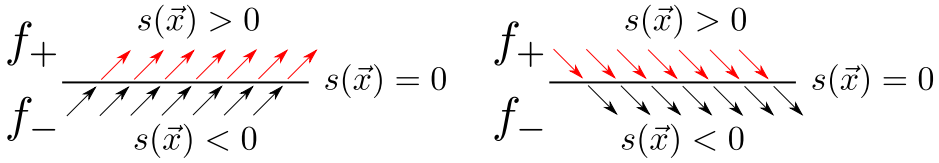
obtaining a system in which the dynamics is everywhere defined (as in Figure 3.7).

$$\dot{\vec{x}} = \begin{cases} f_+(\vec{x}) & \text{if } s(\vec{x}) > 0 \\ f_s(\vec{x}) & \text{if } s(\vec{x}) = 0 \\ f_-(\vec{x}) & \text{if } s(\vec{x}) < 0 \end{cases}$$

Figure 3.7: Switched system with equivalent control on the switching surface.



(a) Stable sliding (b) Unstable sliding



(c) Crossing (d) Crossing

Figure 3.8: Behaviour at the boundary of discontinuity in switched systems.

A sliding motion may either be stable (Figure 3.8a) or unstable (Figure 3.8b); the latter case is of no practical interest because it does not occur in real systems as the solution may leave the boundary of discontinuity at any moment (see e.g. [Fil88, Chapter 2, pp. 51-52]).

As hybrid systems, variable structure systems have a particularly prominent continuous fragment and reasoning about properties of continuous systems is often the dominant component in their analysis. In contrast, computer science interest in hybrid systems began by considering very different systems in which the continuous fragment is much more amenable to formal analysis, such as *timed automata*. Timed automata provide a formal operational model for real-time systems and are, in essence, finite state automata (FSA) augmented with finitely many continuously evolving real *clock* variables, e.g. $\{x_1, \ldots, x_n\}$, evolving according to the differential equation $\dot{x}_i = 1$. Timed automata may impose constraints on the clock variables evolving inside each state and can feature transition *guards* (giving conditions under which a transition is enabled) and *reset maps* that can e.g. reset the value of some clock variables (see Figure 3.9 for an illustration and [AD94, Alu99] for a full formal definition)

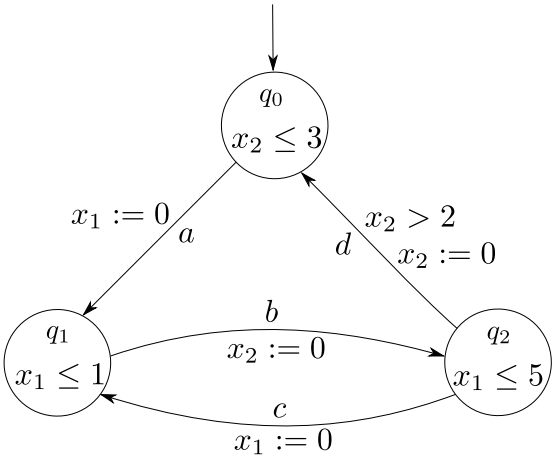


Figure 3.9: Timed automaton with clocks x_1, x_2.

General hybrid dynamical systems generalize both variable structure systems and timed automata by allowing both *reset maps*, which enable the *state* (rather than just the dynamics) to change discontinuously, and the evolution of continuous variables according to general ordinary differential equations. Hybrid systems are known to possess certain properties and exhibit phenomena that do not occur in purely discrete or purely continuous systems. A good overview of hybrid systems may be found in [HLLD09].

3.2.1 Specification formalisms

A popular formalism for specifying hybrid systems is a *hybrid automaton* [ACHH92, Hen96], which may be viewed as a finite state automaton (FSA) augmented with differential equations that govern the system's continuous state evolution while inside a discrete state. The general definition of a hybrid automaton is rather formidable at first sight, but the framework may be appreciated more readily when rendered graphically as a transition system. Below, for completeness of presentation, we reproduce the full formal definition.

Definition 13 *(Hybrid automaton [LJS+03, KGG+09]).* *A hybrid automaton is given by*

$$(Q, Var, \vec{f}, Init, Inv, T, G, R),$$

where

- $Q = \{q_0, q_1, \ldots, q_k\}$ *is a* finite *set of discrete states (modes),*
- $Var = \{x_1, x_2, \ldots, x_n\}$ *is a* finite *set of continuous variables,*
- $f : Q \times \mathbb{R}^n \to \mathbb{R}^n$ *gives the vector field defining continuous evolution inside each mode,*
- $Init \subset Q \times \mathbb{R}^n$ *is the set of initial states,*
- $Inv : Q \to 2^{\mathbb{R}^n}$ *gives the* mode invariants *constraining evolution for every discrete state,*
- $T \subseteq Q \times Q$ *is the transition relation,*
- $G : T \to 2^{\mathbb{R}^n}$ *gives the guard conditions for enabling transitions,*
- $R : T \to 2^{\mathbb{R}^n \times \mathbb{R}^n}$ *gives the reset map.*

A hybrid state *of the automaton is of the form* $(q, \vec{x}) \in Q \times \mathbb{R}^n$.

A *hybrid time trajectory* [LJS+03] is a sequence (which may be finite or infinite) of intervals

$$\tau = \{I_i\}_{i=0}^{N},$$

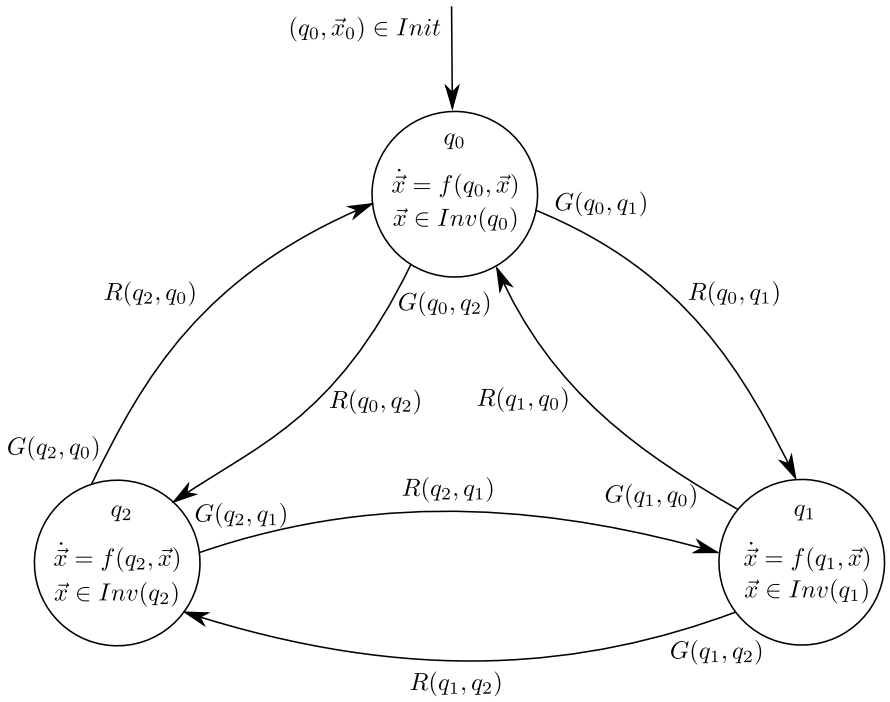


Figure 3.10: Hybrid automaton template with 3 discrete states.

for which $I_i = [\tau_i, \tau_i']$ for all $i < N$ and $\tau_i \leq \tau_i' = \tau_{i+1}$ for all i. If the sequence is finite, then either $I_N = [\tau_N, \tau_N']$ or $I_N = [\tau_N, \tau_N')$. Intuitively, one may think of τ_i as the times at which discrete transitions occur [LJS+03].

An *execution* (or a *run*) of a hybrid automaton defined to be $(\tau, q, \varphi_t^i(\vec{x}))$, where τ is a hybrid time trajectory, $q : \langle\tau\rangle \to Q$ (where $\langle\tau\rangle$ is defined to be the set $\{0, 1, \ldots, N\}$ if τ is finite and $\{0, 1, \ldots\}$ otherwise [LJS+03]) and $\varphi_t^i(\vec{x})$ is a collection of diffeomorphisms $\varphi_t^i(\vec{x}) : I_i \to \mathbb{R}^n$ such that $(q(0), \varphi_0^0(\vec{x})) \in Init$, for all $t \in [\tau_i, \tau_i')$ $\dot{\vec{x}} = f(q(i), \varphi_t^i(\vec{x}))$ and $\varphi_t^i(\vec{x}) \in Inv(i)$. For all $i \in \langle\tau\rangle \setminus \{N\}$ it is also required that transitions respect the guards and reset maps, i.e. $e = (q(i), q(i+1)) \in T$, $\varphi_{\tau_i'}^i(\vec{x}) \in G(e)$ and $(\varphi_{\tau_i'}^i(\vec{x})), \varphi_{\tau_{i+1}}^{i+1}(\vec{x})) \in R(e)$.

Variable structure systems may be modelled as hybrid automata, however the mode invariants (Inv) describing the evolution constraints for each mode need to be such as to allow discrete transitions between the modes. For instance, one cannot simply write down a switched system in Figure 3.7 on page 26 as a hybrid automaton with three discrete states that impose evolution constraints given by $s > 0$, $s = 0$ and $s < 0$. Generally, it is known that modelling variable structure systems using hybrid automata can already be far from trivial [NLC11]. Additionally, problems such as Zeno behaviour (when the system performs

infinitely many discrete transitions in a finite time interval [DHLP09]) present obstacles for simulation and the existence of global solutions (see [DHLP09]). Certain authors choose to altogether ignore Zeno behaviour in defining what constitutes a solution [DHLP09] by only considering non-Zeno runs of the system [Pla10a].

An alternative formalism introduced by Platzer [Pla08] is called a *hybrid program*. It seeks to provide a concise operational model for hybrid systems and enjoys the property of having compositional semantics, which means that proving properties about hybrid programs may be reduced to proving properties about their constituent components [Pla08]. The *transition semantics* of hybrid programs, giving the transition relation on the states, may be found in [Pla10b, Definition 2.7] and [Pla10b, Section 3.3.1], with *trace semantics* given in [Pla10b, Definition 4.3].

Definition 14 *(Hybrid program [Pla08]).* *A hybrid program is defined as follows:*

- $x := \theta$ *: discrete assignment,*
- $x := *$ *: non-deterministic assignment,*
- $x_1' = \theta_1, \ldots, x_n' = \theta_n \;\&\; H$ *: continuous evolution according to an ODE under constraint,*
- $?H$ *: test if formula* H *is true in the current state,*
- $\alpha;\ \beta$ *: sequential composition of hybrid programs* α *and* β*,*
- $\alpha \cup \beta$ *: non-deterministic choice between running hybrid program* α *or* β*,*
- α^* *: non-deterministic (finite) repetition of hybrid program* α *(Kleene star).*

Example 15 *Switched damped oscillator*
The motion of a damped oscillator, such as that shown in Figure 3.11, can be described by the linear second-order differential equation

$$\ddot{x} + 2d\omega\dot{x} + \omega^2 x = 0,$$

where $\omega = \sqrt{\frac{k}{m}}$ is the frequency of oscillation, $d = \frac{c}{2\sqrt{km}}$ is the damping, where c (Nsm^{-2}) is the viscous damping factor, and x is the displacement from the point of equilibrium.

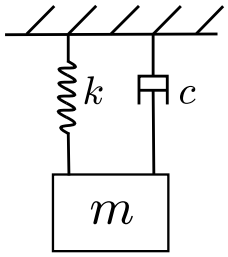


Figure 3.11: Damped oscillator.

We can apply the d'Alembert transformation to convert this into a state space model by setting $x_1 = x$ and $x_2 = \dot{x}$, obtaining

$$\begin{aligned}\dot{x}_1 &= x_2,\\ \dot{x}_2 &= -\omega^2 x_1 - 2d\omega x_2.\end{aligned}$$

Now consider a system in which one can discontinuously change the parameters ω and d between two values (e.g. by successively doubling ω while halving d and halving ω while doubling d) during the evolution. Such a system would induce *mode switching*, which can be modelled using a hybrid automaton (Figure 3.12), or a hybrid program (Figure 3.13).

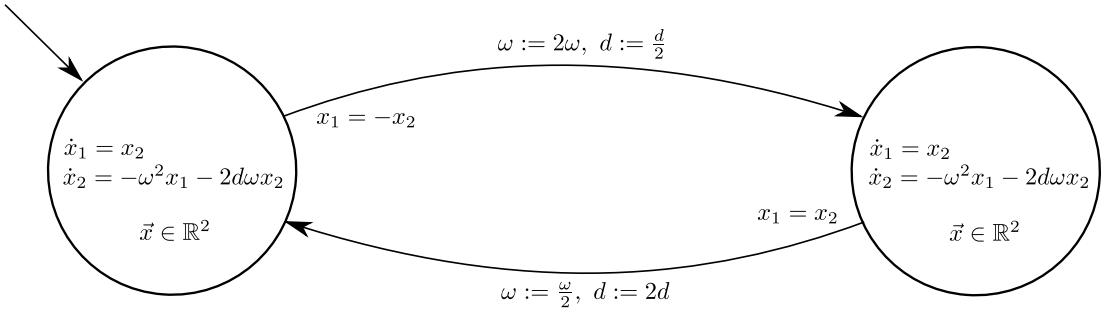


Figure 3.12: Hybrid automaton model of a switched damped oscillator.

Note that in both these models the mode *may* switch whenever a transition guard is enabled. If one wanted to obtain a system in which mode switching *must* happen upon enabling the guard (as in Figure 3.14), one would need to ensure that no further continuous evolution within the current mode is possible; this would require imposing appropriate evolution constraints within each mode. ■

$$
\begin{array}{ll}
\big(& ?(x_1 = -x_2);\ \omega := 2\omega;\ d := d/2 \\
& \cup \\
& ?(x_1 = x_2);\ \omega := \omega/2;\ d := 2d \\
& \cup \\
& \{x_1' = x_2, x_2' = -\omega^2 x_1 - 2d\omega x_2\} \quad \big)^*
\end{array}
$$

Figure 3.13: Hybrid program model of a switched damped oscillator.

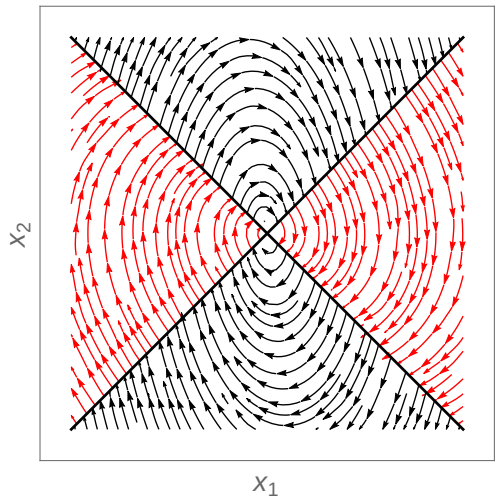


Figure 3.14: Possible phase portrait in a damped oscillator switching between modes where $\omega = 1, d = 0.2$ (red) and $\omega = 2, d = 0.1$ (black).

3.2.2 Verification

Deductive verification of hybrid systems using inductive assertions was explored by Ábrahám-Mumm et al. in [AHS01], employing the general-purpose *PVS* theorem prover [ORS92]. Platzer [Pla08, Pla10a] introduced a powerful verification approach using an extension of dynamic logic, called *differential dynamic logic* (d$\mathcal{L}$), for hybrid programs that provides a specification and verification language for hybrid systems [Pla10a]. Differential dynamic logic extends first-order predicate logic with modalities [ ] and ⟨ ⟩ for hybrid programs that express necessity and possibility. The logic d$\mathcal{L}$ is implemented in *KeYmaera* [PQ08], a theorem prover for hybrid systems that uses hybrid

programs as operational models [Pla11, QML+15]. A typical property one can write down using d$\mathcal{L}$ (see e.g. [QML+15]) is a safety assertion

$$\psi \rightarrow [\ (ctrl;\ plant)^*\ ]\ \phi\ ,$$

where ψ is a formula giving the pre-condition, ϕ is the post-condition and *ctrl* and *plant* are hybrid programs modelling the controller and the plant, respectively, that are run in sequence ($;$) arbitrarily often (*). More recent work by Platzer on a new proof calculus for d$\mathcal{L}$, based on uniform substitution [Pla15], has been implemented in a new theorem prover called *KeYmaera X* by Fulton et al. [FMQ+15].

A popular technique that serves as an alternative to deductive verification for discrete systems is *model checking*. It involves exhaustively exploring all the states visited by the system in order to demonstrate that at no point during the operation is the correctness specification violated. In practice, due to the enormous number of states that arise in real systems (so-called *state explosion problem*), model checking is more useful as a methodology for finding bugs than as a tool for performing verification. The chief practical merit of model checking is its full automation. A user need only appropriately model the system and supply the specification; thereafter, the search for a counterexample is a push-button procedure. Deductive proofs of correctness using an interactive theorem prover, on the other hand, often require significant effort on the part of the user in guiding the proof search.

Remark 16. Much work has been done on improving the scalability of model checking since its inception by Clarke and Emerson [CE82, CES86]. To tackle the state explosion problem, rather than exploring the reachable states explicitly, symbolic model checking [BCM+90] works with symbolic representations of *sets of states* encoded using binary decision diagrams (BDDs). Bounded model checking [BCC+03] exploits recent advances is SAT solving to efficiently explore the reachable states for some bounded time horizon.

Due to the infinite nature of the continuous state space and the complexity of dynamic behaviour, it is in general impossible to compute reachable sets of continuous systems exactly; however, in restricted cases where the continuous dynamics is particularly simple, exact reachable set computation is possible. This was exploited by Henzinger et al., who developed *HyTech* [HHWt97],

a symbolic model checker for *linear hybrid automata* (LHA); that is, hybrid automata in which the continuous evolution is described by differential equations (or inequalities) with *constant* right-hand sides.

Remark 17. Note that even for this simple class of *hybrid* systems, the model checking algorithm is *not* guaranteed to terminate [HHWt97].

In general, continuous systems are impossible to model check because one invariably has to deal with *approximations* of the reachable set. For instance, Frehse et al. extended the approach used in *HyTech* in a tool called *PHAVer* [Fre05] designed for verifying safety in hybrid systems with piecewise-constant derivatives and with the ability to over-approximate *piecewise-affine* continuous dynamics by linear hybrid automata. Other model checking-like approaches have been actively pursued, falling broadly into two different categories:

1. approaches based on computing over-approximations of reachable sets in continuous systems by *abstracting* them with discrete systems [ADI06, TK02, RS07] and
2. methods based on computing bounded-time approximations of the reachable set using *flowpipes*, or *enclosures*, for solutions to ODEs [SDI08, CÁS12, GKC13b, ERNF15, FLGD$^+$11] (also see [BBJ15]).

In the former category, Ratschan developed a tool called *HSolver*, based on abstracting continuous modes by hyper-boxes and using interval arithmetic to analyse the transition behaviour [RS07] of the resulting discrete system. Alur et al. [ADI06] developed methods for abstracting hybrid systems with linear continuous dynamics using linear polynomials. Tiwari and Khanna[TK02] reported an abstraction method for systems with non-linear polynomial dynamics and using non-linear polynomials to perform the abstraction. A common feature of the methods in this class is their ability to prove safety in continuous systems for unbounded time.

In the latter class, Eggers et al. have developed *iSAT-ODE* [ERNF15], a bounded model-checking tool combining enclosure construction for solutions to ODEs (using *VNODE-LP*; see e.g.[Ned06]) with a SAT solver (*iSAT* [FHT$^+$07]). Gao et

al. [GKC13b] also created a tool for bounded model checking of non-linear ODEs, using the functionality provided by the *CAPD* library [Com] for constructing solution enclosures and an SMT solver *dReal* [GKC13a]. Chen et al. created *Flow** [CÁS13], a bounded-time verification tool for hybrid systems using Taylor models [CÁS12] for computing flowpipe approximations of reachable sets in continuous modes. Sankaranarayanan et al. [SDI08] used template polyhedra to construct flowpipe approximations of the reachable set in affine hybrid automata. Frehse et al. developed a verification tool *SpaceEx* [FLGD+11], which is able to work with hybrid systems with affine continuous dynamics and exploits advances in reachable set computation for linear ODEs using support functions [LG09]. Although the underlying algorithms in *SpaceEx* are designed for bounded-time reachability analysis, the tool is able to verify safety properties on unbounded time intervals by successively propagating the reachable set [LG09]; however, termination of this procedure is not guaranteed. Clarke et al. [CFH+03] described an approach that combines discrete abstraction with flowpipes that are polyhedral over-approximations of the reachable set to refine the abstraction. Stursberg and Krogh investigated reachability analysis of hybrid systems using oriented rectangular hulls in [SK03].

Of note also is the work of Asarin [ADG07] and Dang [DGM11], who used a technique known as *hybridization* to approximate non-linear systems by hybrid linear ones, so that verification tools for linear hybrid systems may be applied to the approximation.

It is known that safety verification of hybrid systems reduces entirely to answering questions about the *reachability* of unsafe states [GZ04]. The main problem with model checking-like verification approaches for continuous systems is the nature of the counterexamples they are able to generate: these may no longer be interpreted as witnesses to the reachability of unsafe states. Returning a spurious conuterexample (i.e. one that is not realised by the system) is highly undesirable as it removes one of the key benefits of model checking. Yet, a tool performing reachability analysis in an over-approximation of the continuous dynamics may very well fail to verify safety on the grounds that the specification was violated by the over-approximation of the real system (while the real system may in fact satisfy the specification). In a deductive verification framework, one may reason about reachability in continuous systems by proving certain inductive

invariant assertions, which correspond to over-approximations of reachable sets. The next chapter will give a detailed account of proof methods for checking inductive invariants for continuous systems, reviewing existing techniques and proposing extensions.

In this thesis, we do not view safety verification by model checking the discrete abstraction as any different to a deductive proof of safety. Deductive verification and model checking (in this sense) may be regarded as complementary, rather than competing, approaches to solving the safety verification problem. In Chapter 5 we develop this idea into a method for automatically generating inductive invariants for deductive safety verification that are obtained from reachable sets of discrete semi-algebraic abstractions.

Chapter 4

Safety verification and invariant checking

Synopsis *In this chapter we will (**i**) study the problem of formal safety verification, where one seeks a formal proof that a given continuous system does not evolve into an unsafe state from some given initial configuration. Additionally, we would like to arrive at a proof by working directly with the ODEs, i.e. without solving the initial value problem, as closed-form solutions are often difficult or impossible to find. (**ii**) We will give an extensive survey of existing results on invariant checking, proposing certain extensions, (**iii**) discuss soundness issues in existing approaches and (**iii**) study methods for constructing more efficient proofs of invariance using differential cuts, square-free reduction and order parity decomposition. (**iv**) We introduce a generalisation of so-called* strict barrier certificates *from sub-level sets of polynomials to closed semi-algebraic sets and conclude with a description of a decision procedure for semi-algebraic continuous invariants.*

How to read this chapter

This chapter may be conceptually divided into 4 parts:

1. Introduction to the problem of invariant checking (Section 4.1)
2. Methods for checking invariants described by polynomial equalities (Sections 4.2, 4.3)
3. Methods for checking invariants described by polynomial inequalities (Section 4.4)
4. Methods for checking semi-algebraic invariants (Section 4.5)

The material presented in this chapter is arranged so that the reader may proceed to read it linearly: from the least general class of invariants to the most general.

4.1 Safety verification problem

To state the safety verification problem for continuous systems in its full generality, we require a set of *initial* states for the system, which we denote by $\psi \subseteq \mathbb{R}^n$, and a set of *safe* states denoted $\phi \subseteq \mathbb{R}^n$. The problem is to prove that starting inside ψ, the system $\dot{\vec{x}} = f(\vec{x})$ & H cannot leave ϕ by evolving inside the evolution domain constraint H. We will only consider semi-algebraic ψ and ϕ and will state the safety property formally, using notation from differential dynamic logic (d$\mathcal{L}$) [Pla08], as follows:

$$\psi \rightarrow [\dot{\vec{x}} = f(\vec{x}) \ \& \ H] \ \phi.$$

The above formula asserts that, starting in any state satisfying the pre-condition (ψ), the system will *necessarily* (box modality [ ]) satisfy the post-condition (ϕ) when following the system $\dot{\vec{x}} = f(\vec{x})$ & H for any amount of time. Note that if one considers the continuous system $\dot{\vec{x}} = f(\vec{x})$ & H as a program, the above d$\mathcal{L}$ safety assertion expresses the *Hoare triple* [Hoa69]

$$\{\psi\} \ \dot{\vec{x}} = f(\vec{x}) \ \& \ H \ \{\phi\}.$$

Remark 18. The semantics of the d$\mathcal{L}$ formula $[\dot{\vec{x}} = f(\vec{x}) \ \& \ H] \ \phi$ introduces a *non-deterministic* choice for the (finite) duration of the solution $\varphi_t(\cdot)$ and the box modality [ ] *universally quantifies* over every possible choice of this duration. See [Pla10a] for the full formal definition of the semantics of d$\mathcal{L}$ formulas.

If the solution to the initial value problem for the system $\dot{\vec{x}} = f(\vec{x})$ with initial value $\vec{x}_0$ (we shall always denote this solution by $\varphi_t(\vec{x}_0)$) is available in closed-form for all initial states $\vec{x}_0 \in \psi$, then one can determine whether the system is safe by checking that $\psi \subseteq \phi$ and

$$\forall \vec{x} \in H \cap \psi. \ \forall \tau \geq 0. \ (\forall t \in [0, \tau]. \varphi_t(\vec{x}_0) \in H) \rightarrow \varphi_\tau(\vec{x}_0) \in \phi.$$

Indeed, in performing this check, one explicitly reasons about the forward reachable set of the system, an object central to many approaches to safety verification.

Definition 19 *(Forward reachable set).* *Given a continuous system $\dot{\vec{x}} = f(\vec{x})$ & H and an initial state $\vec{x}_0 \in H$, the* ***forward reachable set*** *of the*

system is defined to be the set

$$\gamma^+(\vec{x}_0, H) \equiv \{\varphi_t(\vec{x}_0) \mid t \geq 0 \text{ s.t. } \forall\ \tau \in [0,t].\ \varphi_\tau(\vec{x}_0) \in H\}.$$

Remark 20. When there is no evolution domain constraint imposed on the system (i.e. $H \equiv \text{True}$), $\gamma(\vec{x}_0, H)$ gives the *forward orbit* of $\vec{x}_0$ under the flow of the ODE.

Generalising this definition to a setting in which there is an initial *set of states*, given by the *pre-condition* $\psi \subseteq H$, let us define the set reachable from ψ to be union of forward reachable sets starting from any initial state in ψ, i.e. let

$$\gamma^+(\psi, H) \equiv \bigcup_{\vec{x}_0 \in \psi} \gamma^+(\vec{x}_0, H).$$

The safety verification problem may now be stated entirely in terms of reachable sets.

Definition 21 *(Safety in terms of forward reachable sets).* *Given a system $\dot{\vec{x}} = f(\vec{x})\ \&\ H$, a set of unsafe states $\neg\phi \subseteq \mathbb{R}^n$ and a set of initial states $\psi \subseteq H$, the system is* ***safe*** *if and only if*

$$\gamma^+(\psi, H) \cap \neg\phi = \emptyset. \tag{4.1}$$

That is, if the forward reachable set of the system initialised in ψ does not intersect the set of unsafe states.

We see that, in principle, when one is handed a safety verification problem for continuous systems, all one has to do is compute the forward reachable set and check for non-intersection with the set of unsafe states. This is indeed all the reasoning that is required to solve the problem; however, performing these two intuitive steps can in practice be very difficult. One faces two major obstacles:

1. Computing the forward reachable set *exactly* is in general not possible,
2. Checking for non-intersection of two sets, even if their descriptions are available in closed form, will also in general not be decidable.

One would ideally hope to apply the intuitive reasoning behind forward reachable sets to arrive at conditions which are still *sufficient* to conclude that the system

is safe and yet *simple* enough to (at least in theory) admit mechanisation. To achieve this for general systems, one is invariably compelled to resort to *over-approximations* of the forward reachable set by sets with simpler descriptions that allow one to check for non-intersection with the set of unsafe states. The most general definition of over-approximation of the forward reachable set that is suitable for this task is due to Platzer and Clarke [PC08], who generalised standard positively-invariant sets to so-called *continuous invariants* for verifying safety of continuous systems under evolution constraints.

Definition 22 *(Continuous invariant).* *A set $I \subseteq \mathbb{R}^n$ is a continuous invariant for $\dot{\vec{x}} = f(\vec{x})$ & H if and only if*

$$\forall\ \vec{x} \in I.\ \forall\ t \geq 0.\ (\forall\ \tau \in [0,t].\ \varphi_\tau(\vec{x}) \in H) \rightarrow (\forall\ \tau \in [0,t].\ \varphi_\tau(\vec{x}) \in I).$$

We may write this invariance assertion as a formula in $\mathrm{d}\mathcal{L}$ *as follows:*

$$I \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ I.$$

Remark 23. Note that the forward reachable set $\gamma^+(\psi, H)$ is in fact the *smallest* continuous invariant containing $\psi \cap H$.

It is immediate that one can re-state the safety verification problem as the problem of determining the existence of a suitable continuous invariant.

Proposition 24 *(Safety verification with continuous invariants).* *Given a continuous system $\dot{\vec{x}} = f(\vec{x})$ & H, a set of safe states $\phi \subseteq \mathbb{R}^n$ and a set of initial states $\psi \subseteq H$, the system is safe if and only if there exists a continuous invariant set $I \subseteq \mathbb{R}^n$ such that*

$$\psi \subseteq I \subseteq \phi$$

That is, if some continuous invariant I separates the forward reachable set of ψ from the set of unsafe states.

Naturally, if one has the exact description of the forward reachable set, one can simply use $I \equiv \gamma^+(\psi, H)$. However, in practice this is rarely possible and one is instead forced to conduct a search for a continuous invariant I with a suitably simple (e.g. semi-algebraic) description and which is furthermore *sufficient* in

the sense that it satisfies the conditions in the above proposition, allowing one to conclude that the system is indeed safe.

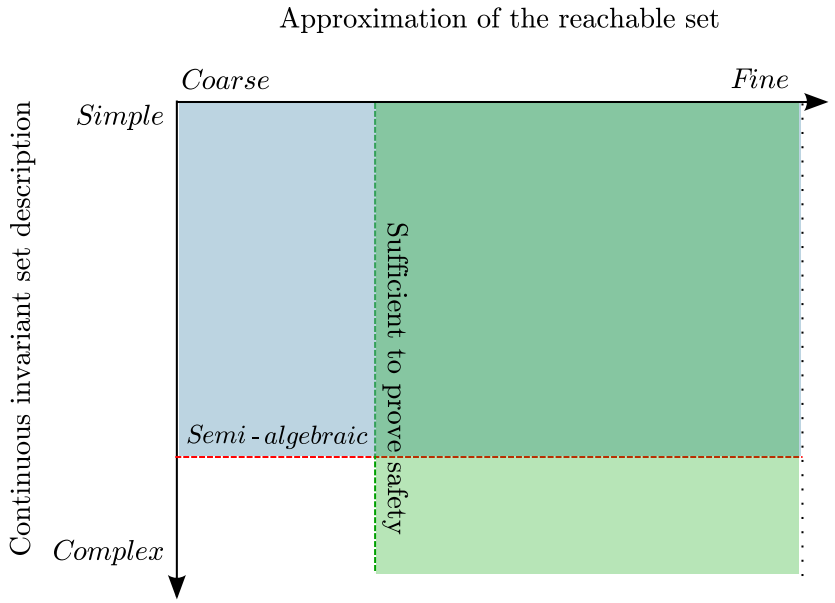


Figure 4.1: Search space for reachable sets in safety verification

Figure 4.1 shows the search space for suitable continuous invariants in a given safe system. The exact description of the forward reachable set is on the far right in the diagram - the complexity of this set depends entirely on the dynamics, but it is always sufficient to conclude safety (provided we are dealing with a safe system). The rectangle to the left of the green dotted line in the diagram represents continuous invarians that are insufficient for proving safety because they include unsafe states; these should be discarded. Continuous invariants below the dashed red line in Figure 4.1 represent sets whose description is too complex to be represented by a finite formula involving polynomial equations and inequalities (semi-algebraic sets) and perhaps even elementary functions ($\sin, \cos, \exp, \log$, etc.). The top right rectangle represents semi-algebraic continuous invariants that are sufficient to prove safety. Effective checks for set membership within this class will be the main focus of this chapter.

In practice, even in the unlikely case when solutions to non-linear ODEs are available explicitly, their form is often much more involved than the differential equations themselves. Transcendental functions, such as sin, cos, exp, log, etc., frequently occur in solutions to ODEs, which means that the description of forward reachable set will find itself below the red dotted line in the diagram.

From a computational point of view, this is highly unfortunate because it introduces a source of undecidability [Ric68] into the verification problem.

Remark 25. In [LPY01] Lafferriere et al. have shown that for linear systems in which the system matrix is diagonalizable with rational eigenvalues one can always explicitly construct the forward reachable set and decide safety by checking for non-intersection with the (semi-algebraic) set of unsafe states.

Using Proposition 24 (even in in the absence of closed-form solutions) the safety verification problem can still sometimes be solved by finding an appropriate continuous invariant, which satisfies the three premises (above the bar) of the following rule of inference:

$$(\mathit{Safety})\ \frac{H \wedge \psi \rightarrow I \qquad I \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\, I \qquad I \rightarrow \phi}{\psi \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\, \phi}$$

In the special case when the invariant is the same as the post-condition, we can drop the last clause and the rule becomes

$$(\text{inv})\ \frac{H \wedge \psi \rightarrow I \qquad I \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\, I}{\psi \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\, I}$$

In the following sections, we will be working with a proof calculus and will call upon this definition in the proofs we construct.

4.1.1 Direct methods for safety verification

An approach to safety verification using continuous invariants is *direct* if it does not require explicit knowledge of the solution, which are typically unavailable in closed form. In what follows we will work exclusively with direct verification methods and will begin by reviewing some fundamental results underlying direct methods for safety verification.

Let us begin with an observation that in the special case when the continuous system $\dot{\vec{x}} = f(\vec{x})$ is not confined to an evolution domain constraint H, the definition of continuous invariant (Definition 22) reduces to the standard definition of positively invariant set from the theory of dynamical systems.

Definition 26 *(Positively invariant set [Bla99]).* *A set $I \subseteq \mathbb{R}^n$ is positively invariant for $\dot{\vec{x}} = f(\vec{x})$ if and only if $\forall\ \vec{x} \in I.\ \forall\ t \geq 0.\ \varphi_\tau(\vec{x}) \in I$. We may write this invariance assertion as a formula in $\mathrm{d}\mathcal{L}$ as $I \rightarrow [\dot{\vec{x}} = f(\vec{x})]\ I$.*

As an immediate consequence of this is the fact, if one succeeds at showing that a given set $I \subseteq \mathbb{R}^n$ is positively invariant for the system $\dot{\vec{x}} = f(\vec{x})$, it is necessarily the case that I is a continuous invariant for the same system under any evolution constraint H. This fact may be concisely summarised in the following sound rule of inference:

$$\frac{\vdash I \rightarrow [\dot{\vec{x}} = f(\vec{x})]\ I}{\vdash I \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ I}\ ,$$

which can be seen to be a special case of the more general rule

$$(\&\ H)\ \frac{H \vdash \hat{H} \quad \vdash I \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ \hat{H}]\ I}{\vdash I \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ I}$$

Necessary and sufficient conditions characterizing *closed* positively invariant sets without requiring explicit solutions to ODEs were first reported in 1942 by Nagumo [Nag42, Bla99]; a result that was later independently re-discovered by many other authors [Yor67, Yor68, Bre70, Cra72, Red72, Har72]. Nagumo's theorem requires the geometric notion of *sub-tangential vectors* to a set.

Definition 27 *(Sub-tangential vector).* *A vector $\vec{v} \in \mathbb{R}^n$ is sub-tangential to a set $S \subseteq \mathbb{R}^n$ at $\vec{x} \in S$ if*

$$\liminf_{\lambda \rightarrow 0^+} \frac{\mathrm{dist}\,(S, \vec{x} + \lambda\vec{v})}{\lambda} = 0,$$

where dist *denotes the Euclidean set distance, i.e.* $\mathrm{dist}(S, \vec{x}) \equiv \inf_{\vec{y} \in S} \|\vec{x} - \vec{y}\|$.

Remark 28. Intuitively, a sub-tangential vector points "inwards" or is tangent to the boundary of the set S.

Theorem 29 *(Nagumo Theorem).* *Given a continuous system $\dot{\vec{x}} = f(\vec{x})$ and assuming that solutions exist and are unique inside some open set $O \subseteq \mathbb{R}^n$, let $S \subset O$ be a closed set. Then, S is* positively invariant *under the flow of the*

system if and only if $f(\vec{x})$ is sub-tangential to S for all $\vec{x} \in \mathrm{bdr}(S)$, where $\mathrm{bdr}(S)$ is the boundary of S.

Proof. See Walter [Wal98, page 117, proof of XVI], or Brezis [Bre70]. □

The statement of this theorem is very general, but the conditions one is required to check to apply the theorem will not, in general, be computable. That being said, many closed sets arising in practice enjoy some "nice" properties which allow one to reason about sub-tangency without the need to apply the basic principles that involve taking limits of the set distance. Among the many independently discovered corollaries to the theorem, the statement given by Bony [Bon69] requires one to check the *sign condition* (i.e. the truth of some predicate with 0 as its second argument, determining the sign of the expression, e.g. $x \leq 0$ or $x_1^2 + x_2^2 + 1 > 0$) of the inner product with the normal vector at the set boundary (see [Red72, Wal98]).

Corollary 1 (Bony, 1969)**.** *Let $\dot{\vec{x}} = f(\vec{x})$ be locally Lipschitz continuous. A closed set $S \subset \mathbb{R}^n$ is positively invariant under the flow of the system if it satisfies the condition*

$$n(\vec{x}) \cdot f(\vec{x}) \leq 0 \quad \textit{for all} \quad \vec{x} \in \mathrm{bdr}(S),$$

whenever $n(\vec{x})$ is a non-zero outer normal *to S at $\vec{x}$ (in the sense of Bony).*

Remark 30. The vector $n(\vec{x})$ is an outer normal to S in the sense of Bony if the open ball $B_{|n(\vec{x})|}(\vec{x} + n(\vec{x}))$ has no points in S [Wal98, Red72]. Note in particular that if S is given by $p \leq 0$ and the boundary $p = 0$ is sufficiently smooth, the (non-zero) gradient $\nabla p(\vec{x})$ may be used in place of the outer normal $n(\vec{x})$ (see Lemma 89 later in this chapter). This property is a key step towards an effective test for sub-tangency, which we will re-visit later in this chapter.

Proof. See [Wal98, page 118], [Red72], or [Bon69] for the original paper (in French). □

Remark 31. Nagumo's theorem is sometimes called the Bony-Brezis theorem, largely because many authors were not aware of Nagumo's existing work, e.g. see [Red72].

In the formal verification community, rather than considering positively invariant sets, a concept similar to the one used in program verification – *inductive invariance* – has been used to reason about safety in continuous systems.

Definition 32 *(Inductive invariant [TT09]).* *Given a continuous system* $\dot{\vec{x}} = f(\vec{x})$ *under no evolution constraints and a set of initial states* $\psi \subseteq \mathbb{R}^n$*, a set* $I \subseteq \mathbb{R}^n$ *is an* ***inductive invariant*** *if* $\psi \subseteq I$ *and*

$$\forall\ \vec{x} \in I.\ \exists\ \tau > 0.\ \forall\ t \in [0, \tau).\ \varphi_t(\vec{x}) \in I.$$

Remark 33. Similar notions have been used elsewhere, e.g. see *local invariance* in [CNV07, Definition 4.1.1].

The inductive invariance property may be used instead of positive invariance to conclude safe operation of a continuous system (for closed sets the two notions coincide provided that solutions are unique).

Proposition 34 *([TT09]).* *If* $I \subseteq \mathbb{R}^n$ *is a closed inductive invariant for the system* $\dot{\vec{x}} = f(\vec{x})$ *such that* $\psi \subseteq I$*, where* ψ *is the set of initial states, then the forward reachable set of the system from* ψ *is contained inside* I*. In particular, taking* $\psi \equiv I$*, we see that* I *is a positively invariant set.*

Proof. See [TT09, Proof of Proposition 5]. □

In order to show inductive invariance of closed sets described by polynomial inequalities, i.e. of the form $I \equiv p \geq 0$, and without solving the system of ODEs explicitly, Taly and Tiwari in [TT09] proposed a direct condition that requires checking sign conditions for infinitely many higher-order Lie derivatives of p, which clearly cannot be computable.

Theorem 35 *(Taly & Tiwari [TT09]).* *Given a set* $I \equiv p \geq 0$ *where* $p \in \mathbb{R}[\vec{x}]$ *and a continuous system* $\dot{\vec{x}} = f(\vec{x})$*,* I *is positively invariant under the flow of the system if and only if for all* $k \geq 1$ *one can show that*

$$p = 0 \rightarrow \bigwedge_{i=1}^{k-1} \mathfrak{L}_f^i(p) = 0 \rightarrow \mathfrak{L}_f^k(p) \geq 0.$$

Proof. See [TT09, Theorem 7 (if), Theorem 8 (only if)]. □

Liu, Zhan and Zhao in [LZZ11] extended the work by Taly and Tiwari and reported a characterisation of positively (inductive) invariant sets under the flow of locally Lipschitz continuous systems of ODEs. In this characterisation, the sets are not required to be be closed nor open. Furthermore, the authors in [LZZ11] reported a direct *decision procedure* for checking semi-algebraic continuous invariants under the flow of polynomial ODEs with semi-algebraic evolution constraints. Ideas similar to [LZZ11] have been developed independently by Ghorbal and Platzer in [GP14a] to give a direct characterisation of algebraic invariants, i.e. inductive invariants of the form $I \equiv p = 0$.

In what follows we will give an extensive survey of existing methods for invariant checking, providing illustrations and highlighting possible issues. The next section will begin with a detailed overview of methods for checking invariance of sets described by polynomial *equalities* (algebraic invariants).

4.2 Checking invariant equations

In this section we will review various computable conditions for checking continuous invariants of the form $I \equiv p = 0$ without explicitly working with the solution $\varphi_t(\cdot)$, before considering the more general problem of semi-algebraic continuous invariant checking in the next section.

In what follows we will present conditions for invariant checking as sound rules of inference in which the conclusion is a formula of the form

$$p = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x}) \ \& \ H] \ p = 0.$$

The premises of the rules will appear as sentences in the decidable theory of real arithmetic.

Whenever possible, we will seek to develop some intuition behind each method, giving illustrative examples where a given method is useful or most natural.

4.2.1 Conserved quantities

An important class of invariants, known as *functional invariants*, *conserved quantities*, or *first integrals*, are real-valued functions on the phase space whose value remains invariant under the flow of the system. The level sets of first integrals define invariant sets for the system.

Remark 36. By *level sets* we understand sets of states where a function evaluates to some fixed value, e.g. $p = 0$ denotes the zero level set of some real-valued function p, i.e. $\{\vec{x} \mid p(\vec{x}) = 0\}$.

Below we give the standard definition used in the theory of dynamical systems.

Definition 37 *(First integral [Gor01]).* *A continuously-differentiable function $I : M \to \mathbb{R}$ defined on some open subset $M \subseteq \mathbb{R}^n$ is a* first integral *for the vector field induced by the system of ODEs $\dot{\vec{x}} = f(\vec{x})$ if and only if $\frac{d}{dt} I(\varphi_t(\vec{x})) = 0$ for all $\vec{x} \in M$.*

First integrals are an important concept in dynamical systems and are central objects in the study of integrability (see e.g. [Gor01]). A typical example of a first integral that is often studied in mechanics is the *Hamiltonian*, which gives the *energy term* in a (Hamiltonian) conservative system.

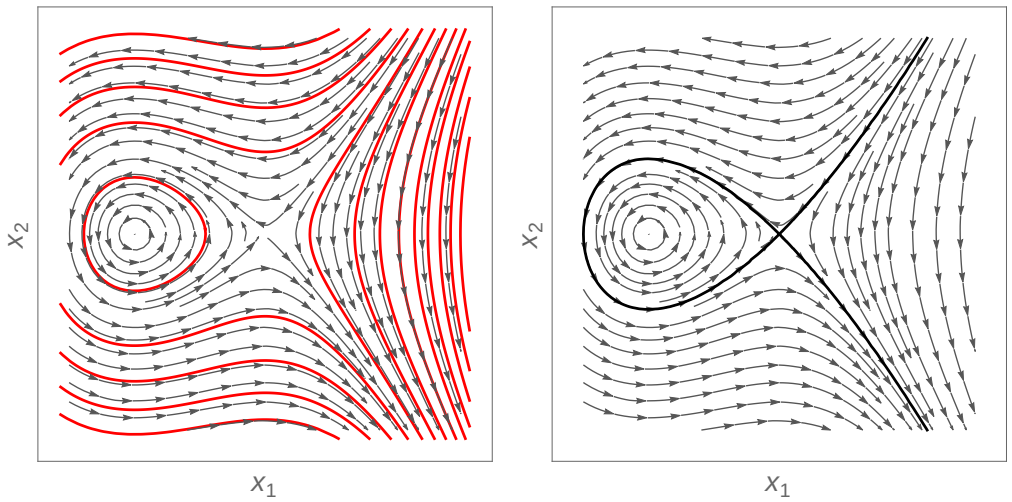
Example 38 *Polynomial first integral*
Consider the conservative non-linear system

$$\begin{aligned} \dot{x}_1 &= -2x_2, \\ \dot{x}_2 &= -3x_1^2 - 2x_1 \end{aligned}$$

and a polynomial function $p(\vec{x}) = x_1^3 + x_1^2 - x_2^2$. Computing the total derivative, we get

$$\begin{aligned} \mathfrak{L}_f(p) &= 3x_1^2\dot{x}_1 + 2x_1\dot{x}_1 - 2x_2\dot{x}_2 \\ &= 3x_1^2(-2x_2) + 2x_1(-2x_2) - 2x_2(-3x_1^2 - 2x_1) \\ &= -2x_2\left(3x_1^2 + 2x_1\right) - 2x_2\left(-3x_1^2 - 2x_1\right) \\ &= 0 \end{aligned}$$

and conclude that $p(\vec{x})$ is a conserved quantity of the system, since its rate of change along the solutions is everywhere zero.



(a) Phase portrait with level sets of the conserved quantity $p(\vec{x})$.

(b) Invariant zero set of the conserved quantity $p(\vec{x}) = 0$.

Figure 4.2: Conservative system.

Figure 4.2 shows the level sets of the conserved quantity $p(\vec{x})$, which define invariant sets for the system. ■

A method for checking invariant equations based on functional invariants was employed for safety verification by Platzer in the d$\mathcal{L}$ verification calculus [Pla10a], where it is known as *differential induction* and (if successful) results in an *equational differential invariant*.

Theorem 39 *(Equational differential invariant).* *The following proof rule is sound:*

$$(\text{DI}_=)\frac{H \vdash \mathfrak{L}_f(p) = 0}{\vdash p = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x}) \ \&\ H]\ p = 0}$$

Proof. See [Pla10a, Theorem 1] (or [Pla10b, Theorem 3.1]). □

Remark 40. Platzer observed in [Pla10b, Proposition 3.4] that a boolean combinations of equational differential invariants can always be reduced (using real arithmetic equivalences $p_1 = 0 \vee p_2 = 0 \equiv_{\mathbb{R}} p_1 p_2 = 0$ and $p_1 = 0 \wedge p_2 = 0 \equiv_{\mathbb{R}} p_1^2 + p_2^2 = 0$) to a single polynomial equality whose invariance can also be proved using the rule $\text{DI}_=$.

4.2.2 Smooth invariant manifolds

A typical example of an invariant set studied in the theory of dynamical systems is an *invariant manifold*. Before we can proceed, we need to recall some basic definitions and theorems from differential geometry that we will also call upon later in this chapter.

Definition 41. *Given a differentiable function $p : \mathbb{R}^n \to \mathbb{R}$, a point $\vec{x} \in \mathbb{R}^n$ for which $\nabla p(\vec{x}) \equiv (\frac{\partial p}{\partial x_1}, \ldots, \frac{\partial p}{\partial x_n})\Big|_{\vec{x}} = \vec{0}$ is called a* singular point *(or* singularity*) of p. Otherwise, if $\nabla p(\vec{x}) \neq \vec{0}$, $\vec{x}$ is known as a* regular point*.*

Definition 42. *A level set of a given differentiable function $p : \mathbb{R}^n \to \mathbb{R}$, $S_k \equiv \{\vec{x} \in \mathbb{R}^n \mid p(\vec{x}) = k\}$ is* regular *if every point on the level set, $\vec{x} \in S_k$, is regular.*

Example 43 *Singularity*
Consider the polynomial $p = x_1^3 + x_1^2 - x_2^2$. Evaluating the gradient at the origin, we obtain

$$\begin{aligned} \nabla p(\vec{0}) &= (3x_1^2 + 2x_1, -2x_2)\big|_{x_1=0, x_2=0} \\ &= \vec{0} \end{aligned}$$

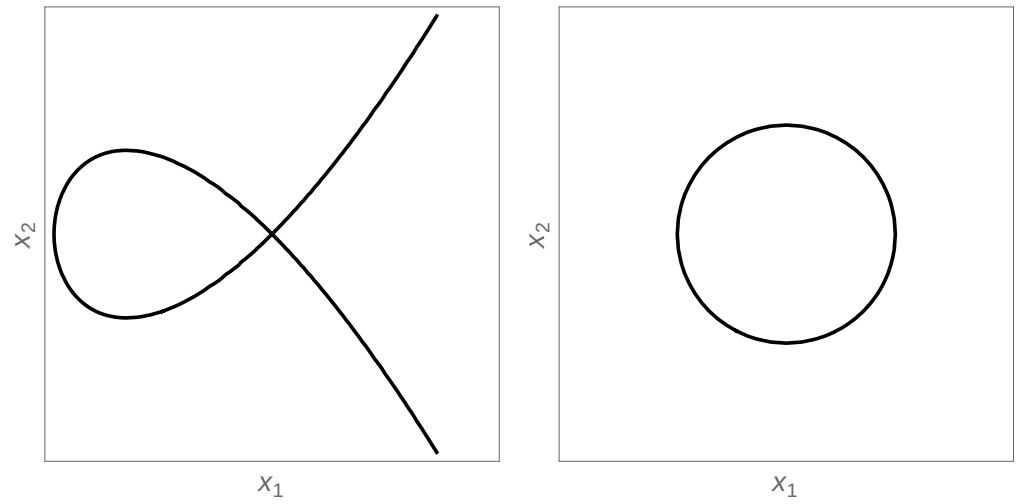
and therefore conclude that there is a singularity at the origin.

Consider now a different polynomial $q = x_1^2 + x_2^2 - 1$, for which $\nabla q(\vec{x}) = (2x_1, 2x_2)$ can only evaluate to the zero vector at the origin. Since $0^2 + 0^2 - 1 \neq 0$, there are no singular points on the zero level set $q = 0$ and it is therefore regular. ■

The most common examples of differentiable manifolds are the familiar $n-$dimensional Euclidean space $\mathbb{R}^n$, as well as open subsets of $\mathbb{R}^n$. The following theorem provides a very convenient (albeit not exhaustive) definition of what constitutes a (sub-)manifold.

Theorem 44 *([Chi06]).* *If $p : \mathbb{R}^n \to \mathbb{R}$ is a C^∞ function, then each of its regular level sets is a sub-manifold of $\mathbb{R}^n$ of dimension $n - 1$.*

Proof. See [Chi06, Proposition 1.92]. □



(a) Singular point at self-intersection in the curve $x_1^3 + x_1^2 - x_2^2 = 0$.

(b) Regular level set $x_1^2 + x_2^2 - 1 = 0$ defining a smooth sub-manifold of $\mathbb{R}^2$.

Figure 4.3: Singular and regular curves.

In working with such (sub-)manifolds, which are also commonly known as *smooth hypersurfaces*, we disregard all level sets with singularities. What we obtain in return for this restriction is a theorem which gives a *necessary and sufficient* condition for invariance of regular level sets. The statement of this theorem requires one further definition.

Definition 45 *(Tangent space).* *If $S \subset \mathbb{R}^n$ is a regular level set defined by $p = 0$, then for any $\vec{x} \in S$, the tangent space to S at $\vec{x}$ is defined to be*

$$T_{\vec{x}}(S) \equiv \{\vec{v} \in \mathbb{R}^n \mid \nabla p(\vec{x}) \cdot \vec{v} = 0\}.$$

More generally, if $p : \mathbb{R}^n \to \mathbb{R}^m$, where $m < n$, and $S \equiv \{\vec{x} \in \mathbb{R}^n \mid p(\vec{x}) = \vec{0}\}$ with the matrix of partial derivatives, i.e. the Jacobian $J_p \equiv (\nabla p_1, \ldots, \nabla p_m)$, having rank m, at all $\vec{x} \in S$, then one defines the tangent space to be all vectors in $\vec{v} \in \mathbb{R}^n$ such that $J_p \vec{v} = 0$. See e.g. [HW96].

Theorem 46 *([Chi06]).* *A sub-manifold S of $\mathbb{R}^n$ is an invariant manifold for the ODE $\dot{\vec{x}} = f(\vec{x})$, if and only if for all $\vec{x} \in S$ we have*

$$(\vec{x}, f(\vec{x})) \in T_{\vec{x}}(S),$$

where $(\vec{x}, f(\vec{x}))$ denotes the Euclidean vector from $\vec{x}$ to $f(\vec{x})$.

Proof. See [Chi06, Proposition 1.107]. □

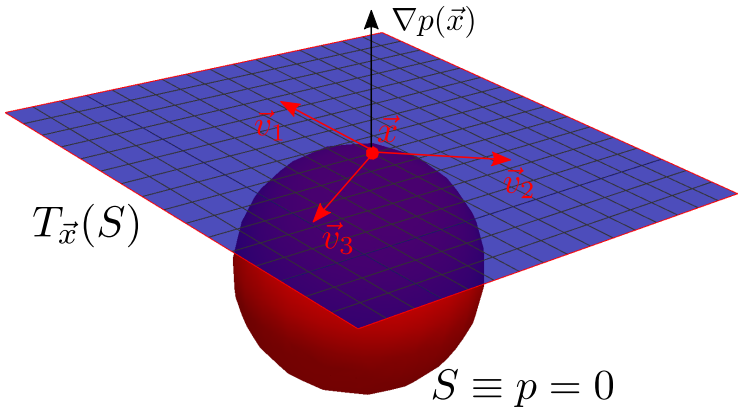


Figure 4.4: Tangent space $T_{\vec{x}}(S)$ to a 2-dimensional sub-manifold S of $\mathbb{R}^3$, given by the regular level set (smooth surface) $p = 0$, at a point $\vec{x}$ and tangent vectors $\vec{v}_1, \vec{v}_2, \vec{v}_3 \in T_{\vec{x}}(S)$.

Whenever $\vec{x}$ is a regular point on the level set $p = 0$ and $\mathfrak{L}_f(p) \equiv \nabla p \cdot f(\vec{x}) = 0$, the vector $(\vec{x}, f(\vec{x}))$ is tangent to the surface and therefore lies in $T_{\vec{x}}(p = 0)$. This observation leads to a necessary and sufficient criterion for checking invariance of smooth hypersurfaces in $\mathbb{R}^n$, first studied by Lie [Lie93] (see Olver [Olv98] and Platzer [Pla12a]).

Proposition 47 *(*Lie*: Smooth equational invariants). The following rule is sound.*

$$\text{(Lie)}\ \frac{p = 0 \vdash \mathfrak{L}_f(p) = 0 \wedge \nabla p \neq \vec{0}}{p = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ p = 0}$$

Proof. See e.g. [Pla12a, Theorem 20]. □

4.2.3 Extending Lie to handle singularities

The proof rule Lie is only able to check invariance of smooth hypersurfaces, which has the consequence that it is unable to e.g. prove invariance of isolated points such as system equilibria for the simple reason that a description of an isolated point $\vec{a} = (a_1, \ldots, a_n) \in \mathbb{R}^n$ is (whenever $n > 1$) given by the zero set of some non-negative real valued function that has $\vec{a}$ as its only global minimum, such as e.g. the sum-of-squares equation

$$p(\vec{x}) = (x_1 - a_1)^2 + \cdots + (x_n - a_n)^2 = 0.$$

This sum-of-squares polynomial p is such that $p(\vec{a}) = 0$ and $p(\vec{x}) > 0$ for all $\vec{x} \in \mathbb{R}^n \setminus \{\vec{a}\}$. At global minima, in this case $\vec{a}$, the gradient of the function vanishes and thus we have $\nabla p(\vec{a}) = \vec{0}$, which violates the regularity (i.e. smoothness) condition in the premise of the proof rule Lie, namely

$$p = 0 \longrightarrow \nabla p \neq \vec{0} \ . \tag{4.2}$$

The condition $p = 0 \rightarrow \mathfrak{L}_f(p) = 0$ is *necessarily true* when $p = 0$ is an invariant equation, i.e. the following is a valid inference

$$\frac{p = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})]\ p = 0}{p = 0 \rightarrow \mathfrak{L}_f(p) = 0} \ .$$

Note that simply removing (4.2) from the premise of the proof rule Lie is unsound (see e.g. [Pla12a, Counterexample 8]); that is, the condition $p = 0 \rightarrow \mathfrak{L}_f(p) = 0$ alone is insufficient to prove the invariance property for $p = 0$, due to potential singularities in the level set $p = 0$. To represent singular points more concisely, we introduce the following definition.

Definition 48 *(Singular Locus).* *Let $p \in \mathbb{R}[x_1, \ldots, x_n]$, the* singular locus *of $p = 0$, henceforth denoted* $\mathrm{SL}(p)$*, is the set of singular points, that is, points $\vec{x} \in \mathbb{R}^n$ satisfying*

$$p = 0 \wedge \frac{\partial p}{\partial x_1} = 0 \wedge \cdots \wedge \frac{\partial p}{\partial x_n} = 0 \ .$$

At singular points, the Lie derivative of p along *any* vector field f reduces to $\nabla p(\vec{x}) \cdot f(\vec{x}) = \vec{0} \cdot f(\vec{x}) = 0$. To avoid these degenerate cases, the regularity condition (4.2) rules out singularities altogether. We now present two extensions of Lie that, in a similar vein to [TT09], seek to partially overcome the strong regularity condition by treating the points on the singular locus separately.

Scenario I: No flow at singularities. Equilibria are points in the state space where the vector field vanishes ($f(\vec{x}) = \vec{0}$) so that there is no motion. However, as seen above, Lie cannot generally be applied to prove invariance properties of isolated equilibria because their description involves singularities. One simple way to resolve this issue is to drop the non-vanishing gradient condition and replace it with the proviso that there be no flow (that is $f(\vec{x}) = \vec{0}$) in the variables of

the invariant candidate on the singular locus; this will allow singularities in the invariant candidate and will provide a *sound* proof method in which there is no need to check for non-vanishing gradient.

Proposition 49 *(*Lie$^\circ$*).* *The following rule is a sound generalisation of* Lie*:*

$$(\text{Lie}^\circ)\frac{p=0 \vdash \mathfrak{L}_f(p)=0 \wedge \big(\text{SL}(p) \to \bigwedge_{x_i \in \text{vars}(p)} \dot{x}_i = 0\big)}{p=0 \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\; p=0}$$

where vars(p) *denotes the set of state variables* x_i *occurring in the polynomial* p*.*

Example 50

Consider again the system from Example 38. The singularity in the curve $x_1^3 + x_1^2 - x_2^2 = 0$ at the origin is an equilibrium of the system. One can thus use Lie$^\circ$ to prove that the curve is a continuous invariant. ■

Scenario II: Flow directed into the level set. The Lie$^\circ$ proof rule can be generalised further at the expense of adding an extra variable by replacing the "no flow" condition ($f_i = 0$) for points on the singular locus with $\forall\lambda.\, p(\vec{x}+\lambda f(\vec{x})) = 0$, where λ is a fresh symbol.

Proposition 51 *(*Lie**).* *The following rule is a sound generalisation of* Lie$^\circ$*:*

$$(\text{Lie}^*)\;\frac{p=0 \vdash \mathfrak{L}_f(p)=0 \wedge (\text{SL}(p) \to p(\vec{x}+\lambda f(\vec{x}))=0)}{p=0 \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\; p=0}$$

To prove soundness of the rules Lie$^\circ$ and Lie*, we appeal to the Nagumo theorem (Theorem 29).

Let us observe that given a closed $S \subset \mathbb{R}^n$ and some $\vec{x} \in \text{bdr}S$, if it is the case that $\vec{x} + \lambda f(\vec{x}) \in S$ for all $\lambda \in \mathbb{R}$, then $\text{dist}\,(S, \vec{x} + \lambda f(\vec{x})) = 0$ and $f(\vec{x})$ is sub-tangent to S at $\vec{x}$. This observation is important for real algebraic sets for which $\text{bdr}S = S$, and the condition $\vec{x}+\lambda f(\vec{x}) \in S$ translates to $p(\vec{x}+\lambda f(\vec{x})) = 0$. This is the main idea behind the soundness of the proof rule Lie*.

Proposition 52. *The proof rule* Lie* *is sound.*

Proof. A point on the variety of p is either regular or singular. For regular points (these form an *open subset* of the variety), since $\mathfrak{L}_f(p(\vec{x})) = 0$, the vector $f(\vec{x})$ is

sub-tangent to the variety at $\vec{x}$ (since it is *tangent* and the condition we check is exactly that which is used in Lie). At singular points $\vec{x} \in \mathcal{V}_{\mathbb{R}}(p)$ if $p(\vec{x}+\lambda f(\vec{x})) = 0$ holds for all λ then $\mathrm{dist}(\mathcal{V}_{\mathbb{R}}(p), \vec{x} + \lambda f(\vec{x})) = 0$ for all λ, from which it follows that $\liminf_{\lambda \to 0^+} \frac{\mathrm{dist}(\mathcal{V}_{\mathbb{R}}(p), \vec{x}+\lambda f(\vec{x}))}{\lambda} = 0$ and thus $f(\vec{x})$ is sub-tangent to $\mathcal{V}_{\mathbb{R}}(p)$ at $\vec{x}$. Assuming solutions exist and are unique, the variety $\mathcal{V}_{\mathbb{R}}(p)$ is positively invariant under the flow of the system $\dot{\vec{x}} = f(\vec{x})$ by the Nagumo theorem. □

The case $f(\vec{x}) = 0$ for all $\vec{x}$ in the singular locus is a special case of the proof rule Lie*. Therefore, the soundness of Lie$^\circ$ is an immediate corollary of 52.

Corollary 2. *The proof rule* Lie$^\circ$ *is sound.*

Remark 53. It is worth remarking that proof rules presented in this section, as well as Lie and $\mathrm{DI}_=$, also work for non-polynomial vector fields and invariant candidates which themselves are not polynomial but sufficiently smooth. However, in such cases the resulting arithmetic may no longer be decidable [Ric68].

Square-free reduction While Lie uses a powerful criterion that captures a large class of practically relevant invariant sets, it will fail for some seemingly simple invariant candidates. For instance, the condition in the premise of Lie will not hold when the goal is to prove that $p = x^2 - 6x + 9 = 0$ is invariant, no matter what vector field one considers. The reason for this is simple: x^2-6x+9 factorises into $(x-3)^2$. The problem here lies in the polynomial p itself, rather than the real variety $\mathcal{V}_{\mathbb{R}}(p)$; in fact, $\mathcal{V}_{\mathbb{R}}(p)$ is exactly the singular locus of p and the proof rule Lie fails because *all* points inside $\mathcal{V}_{\mathbb{R}}(p)$ are singular points. More generally, using the product rule we have

$$\begin{aligned}\nabla p_1^k p_2 \cdots p_l &= p_1^k \nabla p_2 \cdots p_l + p_2 \cdots p_l \nabla p_1^k \\ &= p_1^k \nabla p_2 \cdots p_l + k p_1^{k-1} (p_2 \cdots p_l) \nabla p_1 \\ &= p_1 (p_1^{k-1} \nabla p_2 \cdots p_l + k p_1^{k-2} (p_2 \cdots p_l) \nabla p_1),\end{aligned}$$

which has the consequence that any polynomial p which is not square-free will have vanishing gradient at the real roots of factors with multiplicity greater than 1. One can eliminate such annoying instances by reducing p to square-free form, which is a basic pre-processing step used in computer algebra systems. The

square-free reduction of a polynomial p may be computed as follows:

$$\mathrm{SF}(p) = \frac{p}{\gcd\left(p, \frac{\partial p}{\partial x_1}, \ldots, \frac{\partial p}{\partial x_n}\right)}. \tag{4.3}$$

Intuitively, in performing square-free reduction one hopes to shrink the singular locus of the original polynomial. If $\mathrm{SL}(\mathrm{SF}(p))$ is the empty set (which is the case for $p = x^2 - 6x + 9$ in the example given above), the proof rule Lie applies to $\mathrm{SF}(p)$ but not to p. In general, $\mathrm{SF}(p)$ may satisfy the assumptions of the proof rules Lie$^\circ$ or Lie*, while p might not. Furthermore, it is always sound to conclude that $p = 0$ is invariant from the knowledge that $\mathrm{SF}(p) = 0$ is invariant, since real varieties remain unaltered under square-free reduction of their defining polynomials [CLO10], i.e. $\mathcal{V}_{\mathbb{R}}(p) \equiv \mathcal{V}_{\mathbb{R}}(\mathrm{SF}(p))$. Thus, replacing p with $\mathrm{SF}(p)$ in the premise of Lie, Lie$^\circ$ and Lie* preserves soundness and enlarges the class of polynomials that these proof rules can work with. In addition to increasing the deductive power, square-free reduction reduces the total degree of the polynomials appearing in the premise and thereby reduces the computational complexity of applying the proof rules. It is therefore reasonable to adopt the convention that invariant candidates supplied to Lie and its generalisations are square-free reduced in the premise.

4.2.4 Second integrals and Darboux polynomials

Second integrals are used in the study of integrability of dynamical systems and characterise a more general class of invariants than first integrals.

Definition 54 *(Second integral [Gor01]). A continuously-differentiable function $g : \mathbb{R}^n \to \mathbb{R}$ is a* second integral *for the vector field induced by the system of ODEs $\dot{\vec{x}} = f(\vec{x})$ if*

$$\mathfrak{L}_f(g) = \alpha g$$

for some continuously-differentiable $\alpha : \mathbb{R}^n \to \mathbb{R}$. If g is a constant function, the second integral is trivial.

Remark 55. Note that every first integral is also a second integral, simply by taking α to be 0.

Polynomial second integrals were first studied by Darboux in 1878 [Dar78], who referred to them as *algebraic particular integrals*; today they are also known as *Darboux polynomials* [Gor01, Chapter 2], or *algebraic invariant manifolds* [Gin09, Chapter 6]. To be more explicit, we give the following definition.

Definition 56. *A polynomial $p \in \mathbb{R}[\vec{x}]$ is* Darboux *for the ODE $\dot{\vec{x}} = f(\vec{x})$ if $\mathfrak{L}_f(p) = \alpha p$ for some $\alpha \in \mathbb{R}[\vec{x}]$, which is equivalent to $\mathfrak{L}_f(p) \in \langle p \rangle$.*

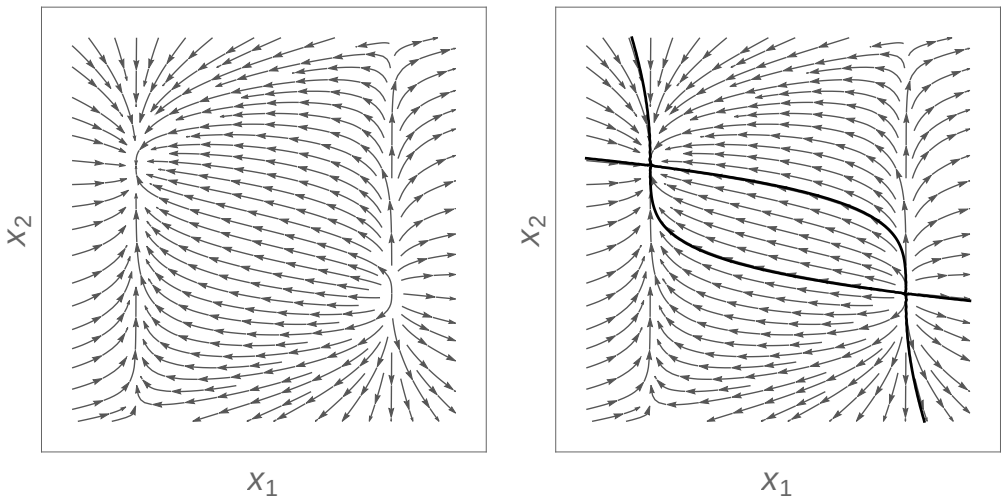
Example 57 *Darboux polynomial*
Consider the non-linear system from [Gor01, Example 2.20] (due to Żołądek):

$$\begin{aligned} \dot{x}_1 &= 3\left(x_1^2 - 4\right), \\ \dot{x}_2 &= 3 - x_2^2 + x_1 x_2 \end{aligned}$$

and let p be the polynomial $x_2^4 + 2x_1x_2^3 + 6x_2^2 + 2x_1x_2 + x_1^2 - 3$. Computing the derivative, we obtain

$$\begin{aligned} \mathfrak{L}_f(p) &= -4x_2^5 - 2x_1x_2^4 + 12x_1^2x_2^3 - 24x_2^3 + 28x_1x_2^2 + 8x_1^2x_2 + 12x_2 + 6x_1^3 - 18x_1, \\ &= \underbrace{(6x_1 - 4x_2)}_{\alpha}\underbrace{(x_2^4 + 2x_1x_2^3 + 6x_2^2 + 2x_1x_2 + x_1^2 - 3)}_{p} \end{aligned}$$

and conclude that $\mathfrak{L}_f(p) \in \langle p \rangle$ and p is a Darboux polynomial for the system.



(a) Phase portrait. (b) Darboux invariant $p(\vec{x}) = 0$.

Figure 4.5: Darboux polynomial.

The zero level set of p defines an invariant set for the system. Furthermore, note that the curve $p = 0$ contains singular points.

■

Conditions similar to those for Darboux polynomials have been used to check *controlled-invariance* (i.e. *viability*) of algebraic hyper-surfaces in polynomial control systems [ZWG10, ZW12]. In the verification community, the ideal membership condition equivalent to that of Darboux polynomials was used to perform algebraic invariant checking in [SSM08], where it is called *polynomial scale (PS) consecution* and a weaker version in which $\alpha \in \mathbb{R}$ is termed *constant scale (CS) consecution.*

Theorem 58 *(Darboux [Dar78], Sankaranarayanan et al. [SSM08]).* *The following proof rule is sound:*

$$\text{(P-c)}\ \frac{\vdash \mathfrak{L}_f(p) \in \langle p \rangle}{\vdash p = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ p = 0}$$

Proof. If $\mathfrak{L}_f(p) \in \langle p \rangle$, then $\mathfrak{L}_f(p) = \alpha p$ for some $\alpha \in \mathbb{R}[\vec{x}]$. Consider the second-order Lie derivative $\mathfrak{L}_f^2(p) = \mathfrak{L}_f(\mathfrak{L}_f(p)) = \mathfrak{L}_f(\alpha p) = \alpha \mathfrak{L}_f(p) + \mathfrak{L}_f(\alpha)p$, using the product rule. When p evaluates to 0, we have $\mathfrak{L}_f(\alpha)p = 0$ and $\alpha \mathfrak{L}_f(p) = \alpha^2 p = 0$, so $p = 0 \implies \mathfrak{L}_f(p) = 0$ and $p = 0 \wedge \mathfrak{L}_f(p) = 0 \implies \mathfrak{L}_f^2(p) = 0$. Similarly for higher-order Lie derivatives, all of which vanish if $p = 0$ is true initially. □

Remark 59. Note that second integrals and Darboux polynomials do not rule out singularities in the level set.

In cases when the polynomial equation of interest may be expressed as a sum-of-squares $p = p_1^2 + p_2^2 + \cdots + p_k^2 = 0$, one can view it as defining a real algebraic variety $\mathcal{V}_{\mathbb{R}}(p) \equiv_{\mathbb{R}} \mathcal{V}_{\mathbb{R}}(p_1, p_2, \ldots, p_k)$. The idea of Darboux polynomials may be extended to handle certain cases where the sum-of-squares polynomial is *not* Darboux, but the variety defines an invariant set for the system (see e.g. [CLPW09, ZW12, ZWG10]). The main observation is that even in cases when $\mathfrak{L}_f(p) \notin \langle p \rangle$, it may still be the case that $\mathfrak{L}_f(p_i) \in \langle p_1, p_2, \ldots, p_k \rangle$ for each $1 \leq i \leq k$, which is sufficient to prove that the variety is invariant under the flow of a polynomial system.

4.2.5 Differential radical invariants

In [GP14a] Ghorbal gave *necessary and sufficient* conditions for invariance of algebraic varieties under the flow of polynomial ODEs. In essence, the conditions extend the idea of Darboux polynomials using the ascending chain property of the Noetherian ring $\mathbb{R}[\vec{x}]$.

Theorem 60 *(Ghorbal [GP14a]).* *The following proof rule is sound:*

$$\text{(DRI)}\ \frac{\vdash N > 0,\ \mathfrak{L}_f^N(p) \in \langle p, \mathfrak{L}_f(p), \ldots, \mathfrak{L}_f^{N-1}(p)\rangle,\ p = 0 \to \bigwedge_{i=1}^{N-1} \mathfrak{L}_f^i(p) = 0}{\vdash p = 0 \to [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ p = 0}$$

Proof sketch. If the ideal membership check in the premise succeeds for some Lie derivative of order N, then all Lie derivatives of orders greater than or equal to N can be expressed as linear combinations of the Lie derivatives of order up to $N-1$. If additionally all the Lie derivatives of order up to $N-1$ evaluate to 0 whenever $p = 0$, then *all* higher-order Lie derivatives evaluate to 0.

See [GP14a] for the original DRI rule and [GP14b] for a proof of its soundness. A more general result for semi-algebraic invariants (implying soundness of DRI) may be found in [LZZ11] along with proofs. □

Example 61
Consider the 3-dimensional non-linear system from [MBK10]:

$$\begin{aligned}\dot{x}_1 &= x_2 + x_3,\\ \dot{x}_2 &= 2x_1x_2 + 2x_1x_3,\\ \dot{x}_3 &= -x_3^3 - x_2x_3^2,\end{aligned}$$

and let $p = (x_1x_3 + x_3 - 1)^2 + (x_2 - x_1^2)^2$. Computing the derivative, we can check (using polynomial reduction and Gröbner bases) that

$$\mathfrak{L}_f(p) = -2x_3\,(x_2 + x_3)\,((x_1 + 1)\,x_3 - 1)^2 \notin \langle p\rangle$$

and thus conclude that p is *not* a Darboux polynomial. However, computing the second-order Lie derivative we obtain

$$\mathfrak{L}_f^2(p) = \underbrace{(x_3\,(3x_2 + 4x_3) - 2x_1)}_{\alpha_1}\underbrace{2x_3\,(x_2 + x_3)\,((x_1 + 1)\,x_3 - 1)^2}_{-\mathfrak{L}_f(p)}.$$

Writing this as

$$\mathfrak{L}_f^2(p) = 0p - \alpha_1 \mathfrak{L}_f(p)$$

we see that $\mathfrak{L}_f^2(p) \in \langle p, \mathfrak{L}_f(p) \rangle$. We can use DRI (with $N = 2$) to prove that $p = 0$ defines an invariant set of the system. ■

DRI leads to a *decision procedure* for checking invariance of real algebraic sets in polynomial vector fields [GP14a]. Thus, theoretically, one can always encode conjunctions of equalities into a single sum-of-squares polynomial equation and check for invariance with DRI. However, an approach due to Ghorbal [GSP14] extends the idea described in the last paragraph of the previous section (again using the ascending chain condition) to give a more efficient decision procedure for checking invariance of real algebraic sets. Given a real variety $\mathcal{V}_{\mathbb{R}}(p_1, p_2, \ldots, p_k)$, the method (called $\text{DRI}_\wedge$) proceeds to check that $\mathfrak{L}_f^N(p_i) \in I$, where

$$I \equiv \langle p_1, p_2, \ldots, p_k, \mathfrak{L}_f(p_1), \mathfrak{L}_f(p_2), \ldots, \mathfrak{L}_f(p_k), \ldots, \mathfrak{L}_f^{N-1}(p_1), \mathfrak{L}_f^{N-1}(p_2), \ldots, \mathfrak{L}_f^{N-1}(p_k) \rangle$$

for some $N > 0$ and for each $1 \leq i \leq k$ and also checks (using a decision procedure for real arithmetic) that the inclusion $\mathcal{V}_{\mathbb{R}}(p_1, p_2, \ldots, p_k) \subseteq \mathcal{V}_{\mathbb{R}}(I)$ holds in order to conclude that the variety $\mathcal{V}_{\mathbb{R}}(p_1, p_2, \ldots, p_k)$ is invariant (see [GSP14] for more details).

Example 62 $\text{DRI}_\wedge$ *(Ghorbal [GSP14])*
Consider the simple system

$$\dot{x}_1 = x_2,$$
$$\dot{x}_2 = x_1$$

and an invariant candidate $x_1 = 0 \wedge x_2 = 0$, which corresponds to $\mathcal{V}_{\mathbb{R}}(x_1, x_2)$. We see that $\mathfrak{L}_f(x_1^2 + x_2^2) = 2x_1x_2 + 2x_2x_1 \notin \langle x_1^2 + x_2^2 \rangle$, and also $\mathfrak{L}_f(x_1) = x_2 \notin \langle x_1 \rangle$ and $\mathfrak{L}_f(x_2) = x_1 \notin \langle x_2 \rangle$. However, we have $\mathfrak{L}_f^2(x_1) = x_1 \in \langle x_1, x_2 \rangle$ and $\mathfrak{L}_f^2(x_2) = x_2 \in \langle x_1, x_2 \rangle$ and $x_1 = 0 \wedge x_2 = 0 \rightarrow x_1 = 0 \wedge x_2 = 0$ is clearly valid. Thus, we conclude that the origin is an invariant using $\text{DRI}_\wedge$ with $N = 2$. ■

4.2.6 Practical performance

It is an interesting question whether using a decision procedure for invariant checking (such as DRI) instead of simpler sufficient conditions comes at the

price of greater computational cost. We have investigated the performance of DRI compared to other sufficient proof rules on classes of invariant checking problems where the respective sufficient proof rules are in fact *necessary and sufficient*. Thus, we have compared the performance of DRI with that of Lie on problems where the invariant candidates are defined by regular level sets of polynomial functions (Figure 4.6). The performance of Lie, Lie$^\circ$ and Lie* was benchmarked against that of DRI on a collection of problems where the invariants were isolated equilibria described by sum-of-squares equations with singularities (in dimensions greater than 1) for which Lie$^\circ$ and Lie* are necessary and sufficient (Figure 4.7). Finally, we compared the performance of DI$_=$ to that of DRI on a set of invariants given by level sets of first integrals, i.e. conserved quantities (Figure 4.8). The graphs show the total number of invariant checking problems (on the horizontal axis) that could be solved by the respective proof rules under some time (shown in log scale on the vertical axis). For these problems we used a 60 second timeout; more details may be found in [GSP15a]. Although the set of problems may not be viewed as representative, we observe the decision procedure performing remarkably well in most of the invariant checking problems. However, the sufficient proof rules can also be seen to out-perform the decision procedure on some of the problems. From our observations, we believe a reasonable proof strategy would begin by applying a decision procedure for algebraic invariant checking first and wait for it to time out (after some specified duration) before attempting a proof using the sufficient proof rules in the hope that they perform better.

4.3 Differential cuts

This section will explore some interesting connections between a proof rule called *differential cut* and concepts from integrability theory of dynamical systems as well embeddings of smooth invariant manifolds, providing geometric intuition that allows using knowledge about the structure of the system more effectively. We will study the problem of proving invariance of sets given by conjunctions of

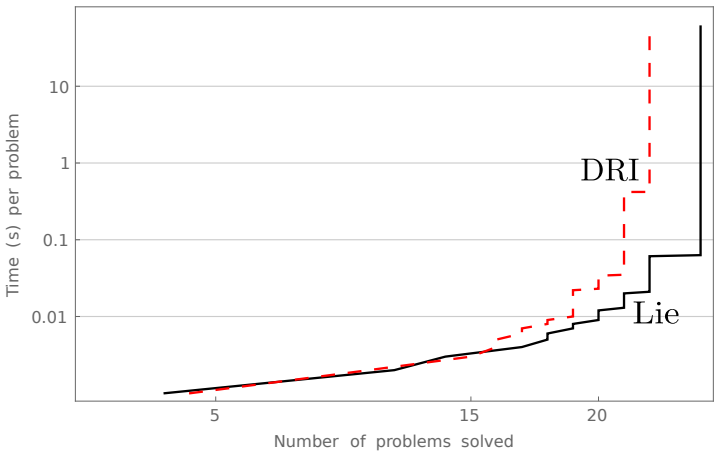


Figure 4.6: Lie vs DRI.

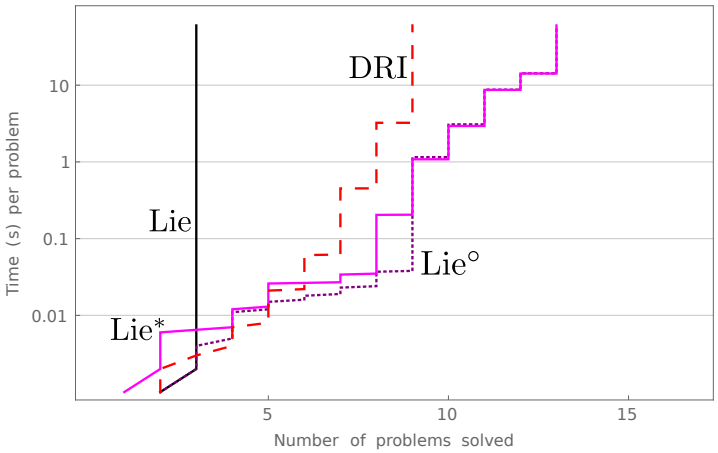


Figure 4.7: Lie, Lie°, Lie* vs DRI.

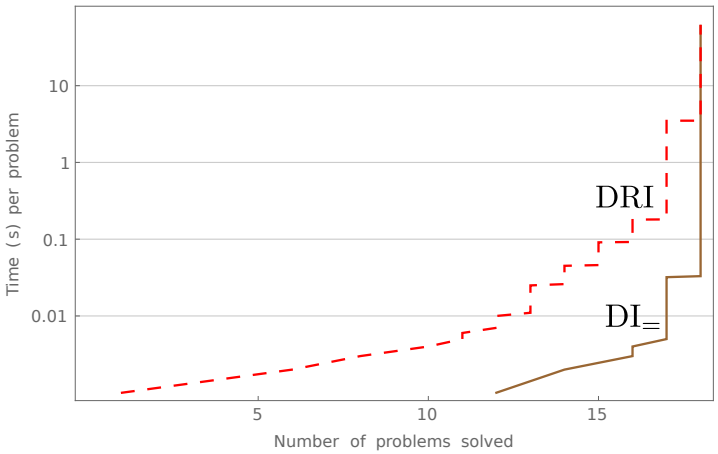


Figure 4.8: DI vs DRI.

polynomial equalities (i.e. real algebraic varieties) and describe a proof strategy using differential cuts which searches for a proof of invariance by decomposing

the conjunction and applying sufficient proof rules.

When considering an invariant candidate that can be expressed as a sum-of-squares equation, one may use a real arithmetic equivalence to reduce it to a conjunction of equalities, i.e.

$$p_1^2 + p_2^2 + \cdots + p_r^2 = 0 \equiv_{\mathbb{R}} p_1 = 0 \wedge p_2 = 0 \wedge \cdots \wedge p_r = 0.$$

It may be the case that each conjunct $p_i = 0$ considered separately defines an invariant for the system. Then, one could simply invoke the following basic result about invariant sets to prove invariance of the conjunction.

Proposition 63. *Let $S_1, S_2 \subseteq \mathbb{R}^n$ be continuous invariants for the continuous system $\dot{\vec{x}} = f(\vec{x})$, then the set $S_1 \cap S_2$ is also a continuous invariant.*

Proof. Elementary. □

Corollary 3. *The proof rule*

$$(\wedge_{\text{inv}})\frac{p_1 = 0 \to [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ p_1 = 0 \qquad p_2 = 0 \to [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ p_2 = 0}{p_1 = 0 \wedge p_2 = 0 \to [\dot{\vec{x}} = f(\vec{x})\ \&H\ ]\ (p_1 = 0 \wedge p_2 = 0)} \tag{4.4}$$

is sound and may be generalised to accommodate arbitrarily many conjuncts.

Of course, one still needs to choose an appropriate proof rule in order to prove invariance of atomic equational formulas.

In general, however, even if the conjunction defines an invariant set, the individual conjuncts need *not* themselves define invariants. If such is the case, one cannot simply break down the conjunctive assertion using the rule $\wedge_{\text{inv}}$ and prove invariance of each conjunct individually. In this section, we explore using a proof rule called *differential cut* (DC) to address this issue.

Differential cuts were introduced as a fundamental proof principle for differential equations by Platzer in [Pla10a] and can be used to (soundly) strengthen assumptions about the system evolution.

Theorem 64 *(Differential Cut [Pla10a]).* *The proof rule*

$$(\text{DC})\frac{F \to [\dot{\vec{x}} = f(\vec{x})]\ C \qquad F \to [\dot{\vec{x}} = f(\vec{x})\ \&\ C]\ F}{F \to [\dot{\vec{x}} = f(\vec{x})]\ F},$$

where C and F denote quantifier-free first-order formulas, is sound.

One may appreciate the geometric intuition behind the rule DC if one realises that the left branch requires one to show that the set of states satisfying C is an invariant for the system initialised in any state satisfying F. Thus, the system does not admit any trajectories starting in F that leave C and hence by adding C to the evolution constraint, one does not restrict the behaviour of the original system.

Differential cuts may be applied repeatedly to the effect of refining the evolution constraint with more continuous invariants. It may be profitable to think of successive differential cuts as showing an *embedding of invariants* in a system. In what follows, we will first develop this intuition for DC when it is used with the rule $\mathrm{DI}_{=}$, drawing similarities to other tools developed in integrability theory of dynamical systems, and then consider the geometric aspect of using DC with the rule Lie.

4.3.1 $\mathrm{DC} + \mathrm{DI}_{=}$. Integrable systems and higher integrals

There is an interesting connection between the use of differential cuts together with the rule $\mathrm{DI}_{=}$ and *integrability* of dynamical systems. To appreciate this, we will require a definition.

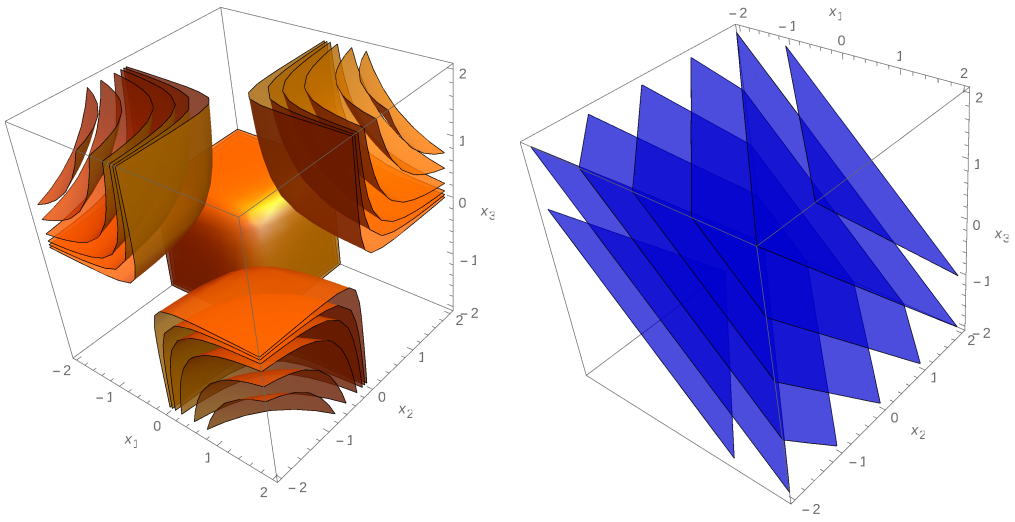
Definition 65 *(Algebraic integrability [Gor01]).* *A system $\dot{\vec{x}} = f(\vec{x})$, where $\vec{x} \in \mathbb{R}^n$ is* ***algebraically integrable***[1] *if there exist $(n-1)$ independent algebraic first integrals p_i, $1 \leq i < n$.*

Example 66 *Integrable system*
Consider the following system from [Gin09, Example 75].

$$\begin{aligned} \dot{x}_1 &= x_1 (x_3 - x_2) , \\ \dot{x}_2 &= x_2 (x_1 - x_3) , \\ \dot{x}_2 &= x_3 (x_2 - x_1) . \end{aligned}$$

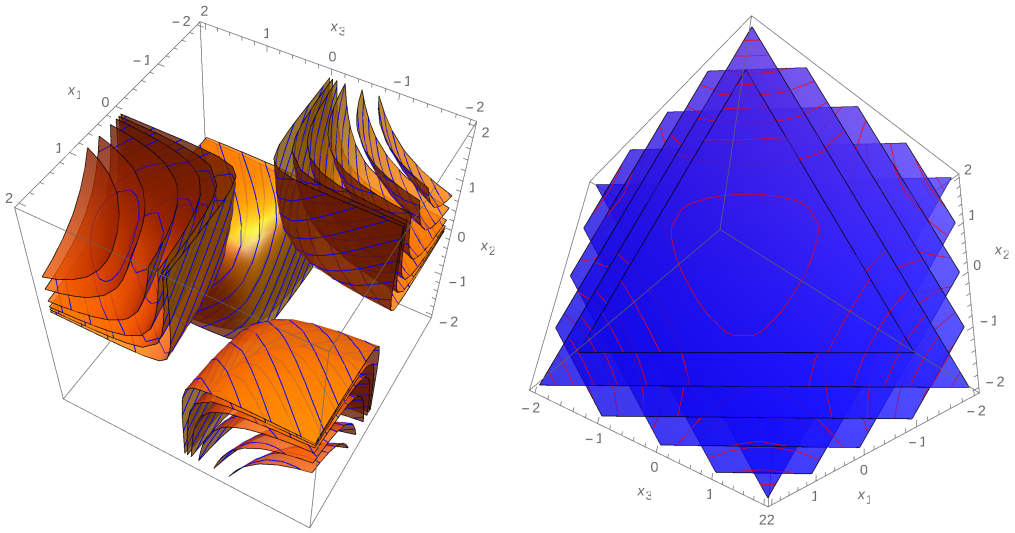
[1] In general, $\vec{x}$ may lie in any affine space $\mathbb{K}^n$, where $\mathbb{K}$ is a field of characteristic zero. See [Gor01].

The system is algebraically integrable because it is 3-dimensional and admits 2 independent polynomial first integrals: $p_1 = x_1x_2x_3$ and $p_2 = x_1 + x_2 + x_3$.



(a) Invariant level surfaces of $x_1x_2x_3$. (b) Invariant level surfaces of $x_1+x_2+x_3$.

Figure 4.9: Foliation of the state space by invariant level surfaces of first integrals.



(a) Invariant level curves of $x_1 + x_2 + x_3$ on invariant level surfaces of $x_1x_2x_3$.

(b) Invariant level curves of $x_1x_2x_3$ on invariant level surfaces of $x_1 + x_2 + x_3$.

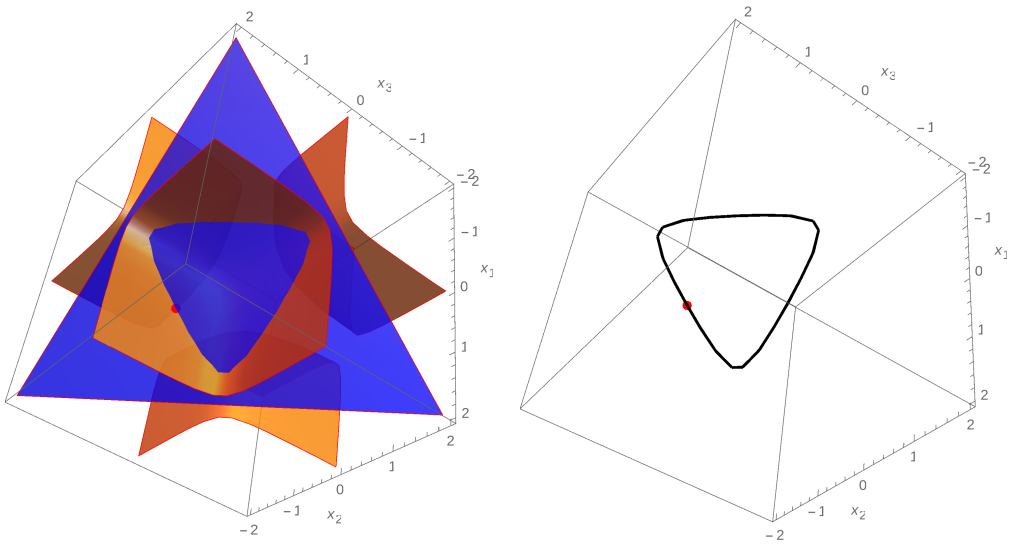
Figure 4.10: Foliation of invariant surfaces by level curves of algebraic first integrals.

Thus, every level set of p_1 and p_2 defines an invariant set for the system and one can conclude that $p_1 = C_1 \wedge p_2 = C_2 \equiv_{\mathbb{R}} (p_1 - C_1)^2 + (p_2 - C_2)^2 = 0$ is also an invariant for arbitrary $C_1, C_2 \in \mathbb{R}$.

Suppose we are given an initial condition $\vec{x}_0 = (1, \frac{1}{16}, 1) \in \mathbb{R}^3$. Taking this point, we can evaluate the two independent first integrals $p_1(\vec{x}_0) = \frac{1}{16}$ and $p_2(\vec{x}_0) = \frac{33}{16}$, obtaining two constants $C_1 = \frac{1}{16}$ and $C_2 = \frac{33}{16}$, respectively. With this, we can construct a *phase curve* which is guaranteed to contain the orbit of the system through the point $\vec{x}_0$; this curve corresponds to the conjunction

$$x_1 x_2 x_3 = \frac{1}{16} \wedge x_1 + x_2 + x_3 = \frac{33}{16}.$$

We should note that in this example it was crucial that we deal with *independent* first integrals, i.e. the system was indeed integrable. This method will not work if one has first integrals such as e.g. p_1 and p_1^2 because one can no longer reduce the dimension of the invariant set defined by the conjunction.



(a) Invariant level sets of first integrals $x_1 x_2 x_3 = \frac{1}{16}$ and $x_1 + x_2 + x_3 = \frac{33}{16}$.

(b) Invariant space curve containing the orbit through $\vec{x}_0 = (1, \frac{1}{16}, 1)$.

Figure 4.11: Invariant space curve construction in an integrable system.

■

Whenever a system is algebraically integrable, with independent first integrals $p_1, p_2, \ldots, p_{n-1}$, and the pre-condition ψ is given by an initial point $\vec{x}_0 \in \mathbb{R}^n$, one can employ differential cuts to successively refine the evolution domain with invariant level sets $p_i(\vec{x}) = C_i$ for any $1 \leq i < n$ where $C_i = p_i(\vec{x}_0)$ until one arrives at a constraint

$$H \wedge p_1 = C_1 \wedge p_2 = C_2 \wedge \cdots \wedge p_{n-1} = C_{n-1}$$

and can conclude system safety by checking that the new domain constraint does not contain any unsafe states.

Alternatively, instead of refining the domain constraint with differential cuts, one can construct equational invariants from first integrals directly using the following theorem (which is perhaps more efficient, but obscures the underlying geometric intuition).

Theorem 67 *(Platzer [Pla10b]).* *A formula F, given by a propositional combination of equalities of the form $p_i = 0$, defines an invariant for the continuous system if all of its equational atoms p_i are first integrals for the system.*

Proof. See [Pla10b, Proof of Proposition 3.4]. □

In [Pla12b] Platzer has shown that differential cuts add to the deductive power of differential induction (this is written as $\mathrm{DC} + \mathrm{DI}_{=} \succ \mathrm{DI}_{=}$ for the equational case). This fact implies that certain invariants can be proved using the rule $\mathrm{DI}_{=}$ in concert with DC, when no such proof is possible without using DC (refuting an earlier hypothesis in [Pla10b]). In practice, what this result shows is that differential cuts together with $\mathrm{DI}_{=}$ allow one to work with more than just the first integrals of the system. Indeed, there is an intriguing correspondence between such proofs and a concept from integrability theory of dynamical systems known as *higher integral* [Gor01].

Recall that the premise of the rule $\mathrm{DI}_{=}$ establishes that $p(\vec{x})$ is a *first integral* (i.e. a constant of motion) for the system in order to conclude that $p = 0$ defines an invariant set. More generally, $p(\vec{x})$ is a *second integral* if $\mathfrak{L}_f(p) = \alpha p$, where α is some function; this is also sufficient to conclude that $p = 0$ is an invariant. If $p \in \mathbb{R}[\vec{x}]$, then one has a *Darboux polynomial* [Gor01, Dar78] and the condition corresponds to the premise of the rule P-c. A *third integral* is a function $p(\vec{x})$ that remains constant on some level set of a first integral $g(\vec{x})$ [Gor01, Section 2.6], i.e. $\mathfrak{L}_f(p) = \alpha g$ where g is a first integral and α is some function. These ideas generalise to *higher integrals*, which are functions that vanish on level sets of other (lower) integrals (see [Gor01, Section 2.7]).

Higher integrals are powerful because they can be used to e.g. show invariance of certain sets in cases when a system may *not* be integrable. The example below

was reported by Platzer in [Pla12a] as a demonstration of the increased deductive power afforded by DC when used together with $\text{DI}_=$. As we shall see, the invariant of interest is in fact a level set of a third integral for the system.

Example 68 *Platzer [Pla12a], deconstructed aircraft. Differential cut with* $\text{DI}_=$*.* Consider an example of the use of $\text{DI}_=$ with DC due to Platzer [Pla12a], where the dynamics is given by the linear system

$$\dot{x}_1 = -x_2,$$
$$\dot{x}_2 = x_3,$$
$$\dot{x}_3 = -x_2,$$

with an invariant candidate $x_1^2 + x_2^2 = 1 \wedge x_3 = x_1$. One cannot use $\text{DI}_=$ directly to prove the goal

$$x_1^2 + x_2^2 = 1 \wedge x_3 = x_1 \rightarrow [\dot{\vec{x}} = f(\vec{x})]\ (x_1^2 + x_2^2 = 1 \wedge x_3 = x_1)\ .$$

We can apply DC to cut by $x_1 = x_3$, which is a first integral for the system and is thus provable using $\text{DI}_=$. The left branch of DC is proved as follows:

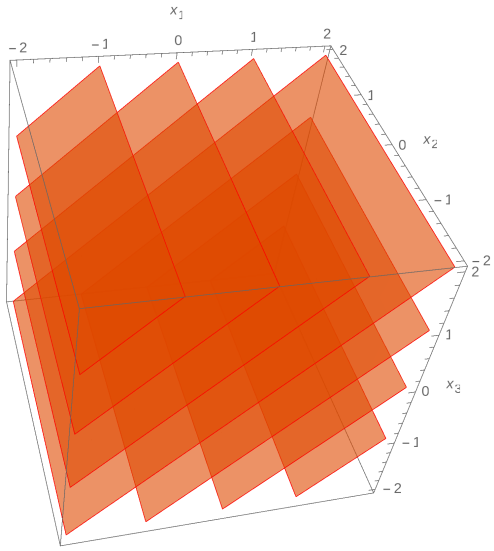
$$\text{(inv)}\ \dfrac{\text{(}\mathbb{R}\text{)}\ \dfrac{*}{x_1^2 + x_2^2 = 1 \wedge x_3 = x_1 \rightarrow x_3 = x_1} \qquad \text{(DI}_=\text{)}\ \dfrac{\text{(}\mathbb{R}\text{)}\ \dfrac{*}{-x_2 = -x_2}}{x_3 = x_1 \rightarrow [\dot{\vec{x}} = f(\vec{x})]\ x_3 = x_1}}{x_1^2 + x_2^2 = 1 \wedge x_3 = x_1 \rightarrow [\dot{\vec{x}} = f(\vec{x})]\ x_3 = x_1}$$

For the right branch of DC we need to show that $x_1^2 + x_2^2 = 1$ is an invariant under the evolution constraint $x_3 = x_1$. This is again provable using $\text{DI}_=$:

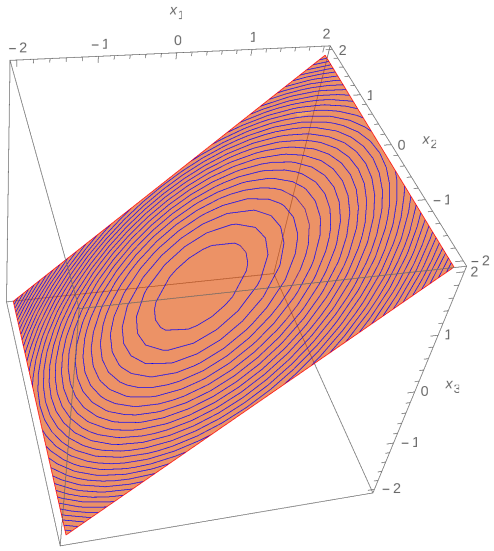
$$\text{(}\wedge_{\text{inv}}\text{)}\ \dfrac{\text{(DW)}\ \dfrac{*}{x_3 = x_1 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ x_3 = x_1]\ x_3 = x_1} \qquad \text{(DI}_=\text{)}\ \dfrac{\text{(}\mathbb{R}\text{)}\ \dfrac{*}{x_3 = x_1 \vdash -2x_1x_2 + 2x_2x_3 = 0}}{x_1^2 + x_2^2 = 1 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ x_3 = x_1]\ x_1^2 + x_2^2 = 1}}{x_1^2 + x_2^2 = 1 \wedge x_3 = x_1 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ x_3 = x_1]\ (x_1^2 + x_2^2 = 1 \wedge x_3 = x_1)}$$

We can now construct a proof of invariance for the conjunction using DC.

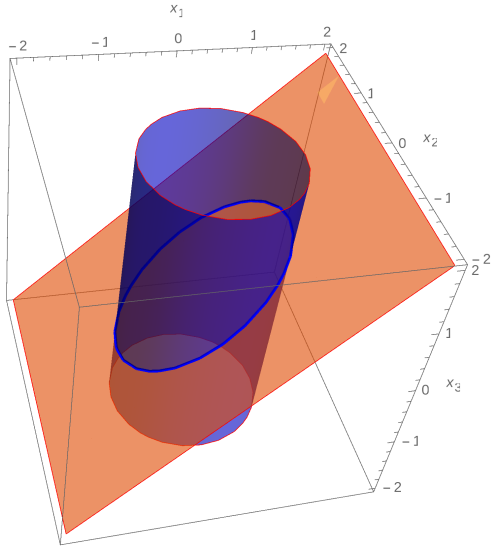
Note that in this example, the rule $\text{DI}_=$ was all that was required to complete the proof of invariance. By first showing that $x_3 - x_1$ is an invariant function (first integral) for the system and restricting the evolution domain to the zero set of the first integral, $x_3 - x_1 = 0$, one can prove that the polynomial $x_1^2 + x_2^2 - 1$ is conserved in the constrained system. In this example we have $\mathfrak{L}_f(x_1^2 + x_2^2 - 1) =$



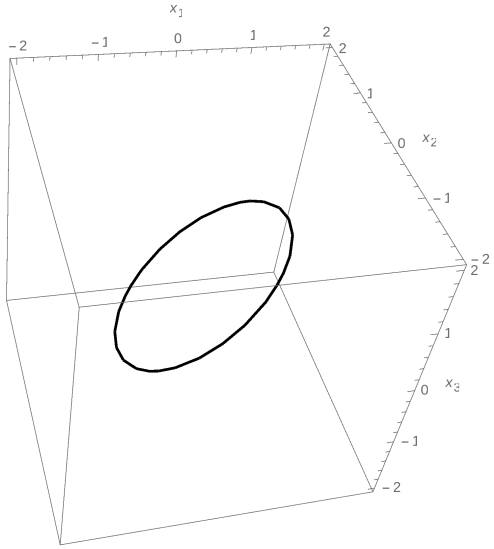
(a) Invariant level sets of the first integral $x_3 - x_1$ foliating the state space of the system.



(b) Invariant level curves of the third integral $x_1^2 + x_2^2 - 1$ foliating *one* level surface of the first integral, $x_3 - x_1 = 0$.



(c) Intersection of the level sets defining the invariant $x_1^2 + x_2^2 = 1 \wedge x_3 = x_1$.



(d) Invariant space curve giving a periodic orbit of the system.

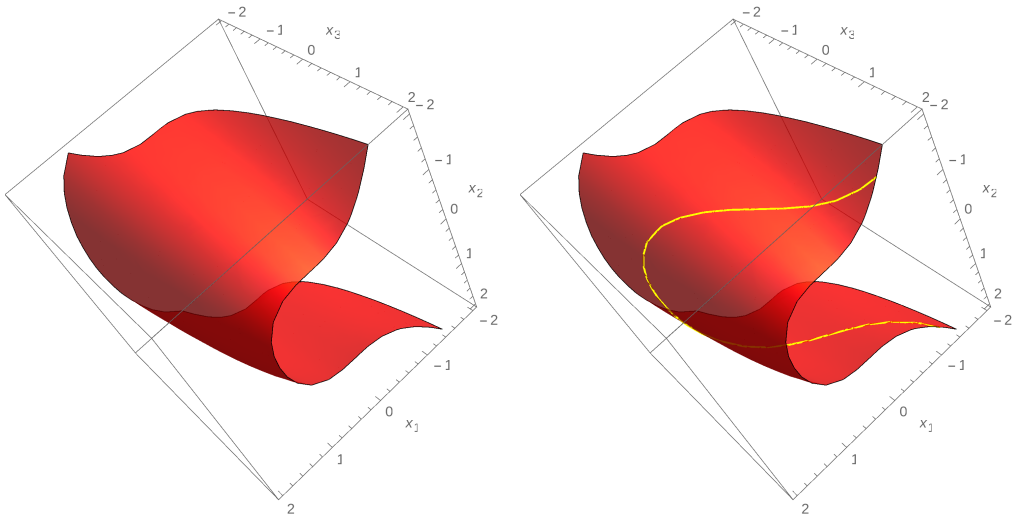
Figure 4.12: Third integral example.

$-2x_1x_2 + 2x_2x_3 = 2x_2(x_3 - x_1)$, where $(x_3 - x_1)$ is a first integral of the system. Thus, $x_1^2 + x_2^2 - 1$ is in fact a (polynomial) third integral.

The geometric intuition behind the use of a third integral in this proof is given in Figure 4.12. ■

4.3.2 DC+Lie. Embeddings of invariant sub-manifolds

Differential cuts are related to embeddings of invariant sub-manifolds, when used with the proof rule Lie. To develop this idea, let us remark that if one succeeds at proving invariance of some $p_1 = 0$ using the rule Lie in a system with no evolution constraint, one shows that $p_1 = 0$ is a smooth invariant sub-manifold of $\mathbb{R}^n$. If one



(a) Smooth manifold $M_1 \subset \mathbb{R}^3$. (b) Sub-manifold $M_2 \subset M_1$ (yellow).

Figure 4.13: Embedding of smooth manifolds.

now considers the system evolving inside that invariant manifold and finds some $p_2 = 0$ which can be proved to be invariant using Lie with $p_1 = 0$ acting as an evolution constraint, then inside the manifold $p_1 = 0$, $p_2 = 0$ defines an invariant sub-manifold (even in cases when $p_2 = 0$ might not define a sub-manifold of the ambient space $\mathbb{R}^n$). One can proceed using Lie in this way to look for further embedded invariant sub-manifolds. We will illustrate this idea using a basic example.

When considering evolution domain constraints H, if H is an open set, one may add it to the context in the premise of Lie (see [Theorem 2.8][Olv98]) to obtain

$$(\text{Lie})\ \frac{H \vdash p = 0 \rightarrow \mathfrak{L}_f(p) = 0 \wedge \nabla p \neq \vec{0}}{\vdash p = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ p = 0}.$$

If H is not open, one needs to ensure that H has a structure of a differentiable manifold in order to add it to the assumptions without losing soundness. In the

case where H is given by a conjunction of polynomial equalities, one may do this by requiring that the matrix of partial derivatives is full rank, i.e. the rule may be stated as

$$\text{(Lie)}\ \frac{\bigwedge_{i=1}^{k} p_i = 0 \vdash p = 0 \rightarrow \mathfrak{L}_f(p) = 0 \wedge rk(\nabla p_1, \ldots, \nabla p_k, \nabla p) = k+1}{\vdash p = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ \bigwedge_{i=1}^{k} p_i = 0]\ p = 0},$$

which reduces to

$$\text{(Lie)}\ \frac{\vdash p = 0 \rightarrow \mathfrak{L}_f(p) = 0 \wedge \nabla p \neq \vec{0}}{\vdash p = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})]\ p = 0}$$

when $k = 0$.

Example 69 *Differential cut with* Lie

Let the system dynamics be given by the simple linear system

$$\dot{x}_1 = x_1,$$
$$\dot{x}_2 = -x_2.$$

This system has an equilibrium at the origin, i.e. $f(\vec{0}) = \vec{0}$. Consider an invariant candidate $x_1 = 0 \wedge x_1 - x_2 = 0$. One cannot use Lie directly to prove the goal

$$x_1 = 0 \wedge x_1 - x_2 = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})]\ (x_1 = 0 \wedge x_1 - x_2 = 0).$$

Instead, DC can be used to cut by $x_1 = 0$, which is an invariant for this system provable using Lie. The left branch of DC is proved as follows:

$$\dfrac{\text{(}\mathbb{R}\text{)}\ \dfrac{*}{x_1 = 0 \wedge x_1 - x_2 = 0 \rightarrow x_1 = 0}\quad \text{(Lie)}\ \dfrac{\text{(}\mathbb{R}\text{)}\ \dfrac{*}{x_1 = 0 \rightarrow x_1 = 0 \wedge (1 \neq 0)}}{x_1 = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})]\ x_1 = 0}}{x_1 = 0 \wedge x_1 - x_2 = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ x_1 = 0]\ x_1 = 0}\ \text{(inv)}$$

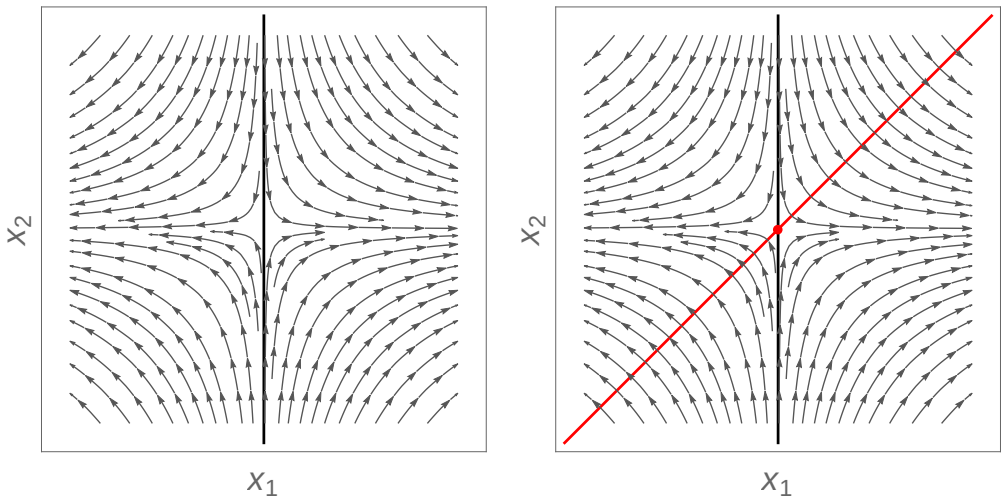
One can also prove that $x_1 = x_2$ is a invariant under the evolution constraint $x_1 = 0$:

$$\dfrac{\text{(DW)}\ \dfrac{*}{x_1 = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ x_1 = 0]\ x_1 = 0}\quad \text{(Lie)}\ \dfrac{\text{(}\mathbb{R}\text{)}\ \dfrac{*}{x_1 = 0 \vdash x_1 - x_2 = 0 \rightarrow x_1 + x_2 = 0 \wedge rk((1,0),(1,-1)) = 2}}{x_1 - x_2 = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ x_1 = 0]\ x_1 - x_2 = 0}}{x_1 = 0 \wedge x_1 - x_2 = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ x_1 = 0]\ (x_1 = 0 \wedge x_1 - x_2 = 0)}\ \text{(}\wedge_{\text{inv}}\text{)}$$

Using these two sub-proofs to close the appropriate branches, the rule DC proves

$$x_1 = 0 \wedge x_1 - x_2 = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ ]\ (x_1 = 0 \wedge x_1 - x_2 = 0).$$

While this example is very simplistic, it provides a good illustration of the method behind differential cuts. We used DC to restrict system evolution to an invariant



(a) Invariant manifold $x_1 = 0$. (b) Sub-manifold at the origin.

Figure 4.14: Differential cut used to show that the intersection at the origin is an invariant sub-manifold of $x_1 = 0$.

manifold $x_1 = 0$ using Lie and then used Lie again to show that $x_1 - x_2 = 0$ defines an invariant sub-manifold inside $x_1 = 0$. This is illustrated in Figure 4.14 on page 72.

It is also worth noting that the choice of conjunct for use in the differential cut was crucial. Had we initially picked $x_1 - x_2 = 0$ to act as C in DC, the proof attempt would have failed, since this does not define an invariant sub-manifold of $\mathbb{R}^2$ (see Figure 4.14 on page 72). ■

Note that by employing DC, we proved invariance of a conjunction which could not be described by an atomic equational assertion which is provable using the rule Lie, or by using Lie to prove invariance of each conjunct after breaking down the conjunction with the rule $\wedge_{\text{inv}}$. Alternatively, one may work with the entire conjunction directly using the following proof rule.

Theorem 70. *The following proof rule is sound:*

$$(\text{Lie}_\wedge)\ \frac{\bigwedge_{i=1}^{k} p_i = 0 \vdash \bigwedge_{i=1}^{k} \mathfrak{L}_f(p_i) = 0 \wedge rk(\nabla p_1, \ldots, \nabla p_k) = k}{\vdash \bigwedge_{i=1}^{k} p_i = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ \bigwedge_{i=1}^{k} p_i = 0}$$

Proof. [Olv98, Theorem 2.8]. □

In practice, checking the full rank condition for a matrix of partial derivatives of polynomial functions (though possible) is very expensive. For instance, suppose one has 2 polynomials $p_1, p_2 \in \mathbb{R}[x_1, x_2]$ and one wishes to check that the matrix $(\nabla p_1, \nabla p_2)$ has full rank everywhere on $p_1 = 0 \wedge p_2 = 0$. One can check that the sentence

$$\forall\, x_1, x_2.\ p_1 = 0 \wedge p_2 = 0 \rightarrow \forall\, \lambda \neq 0.\ \nabla p_1 \neq \lambda \nabla p_2$$

is true over the reals. However, this comes at the price of introducing a fresh variable, and using a decision procedure for (the universal fragment of) real arithmetic, which has time complexity exponential in the number of variables. One can see why the more general variant of Lie that allows assuming H in the premise, and $\text{Lie}_\wedge$, have little practical appeal.

4.3.3 Proof strategies using differential cuts

Differential cuts can be used to search for a proof of invariance of conjunctive equational assertions. This involves selecting some conjunct $p_i = 0$ to cut by (that is use it as C in DC). If the conjunct is indeed an invariant, it will be possible to strengthen the evolution domain constraint and proceed in a similar fashion by selecting a new C from the remaining conjuncts until a proof is attained. A formal proof of invariance using differential cuts can be quite long and will repeatedly resort to proof rules such as $(\wedge_{\text{inv}})$ and DW, which is used to prune away conjuncts that have already been added to the evolution domain constraint.

A simple proof strategy may be implemented as a recursive function to iteratively select a conjunct with which to attempt a differential cut. Before calling this function, the conjuncts could, for instance, be put into ascending order with respect to the number of variables appearing in the conjunct. For purely polynomial problems, it would also be reasonable to make the ordering ascending with respect to the total degree of the polynomials. The aim of such a pre-processing step would be to ensure that those conjuncts which are potentially less expensive to check for invariance are processed first. There is in general no easy way of selecting the "right" proof rule for showing invariance of atomic equations; a possible, albeit not very efficient, solution would be to iterate through all the available proof rules. This would certainly combine their deductive power, but could also lead do diminished performance. In practice, selecting a good

proof rule for atomic invariants is very much a problem-specific matter. The overall proof strategy, if successful, would result in a proof tree resembling that shown in Figure 4.15 on page 75. The proof steps labelled with ? mark choices in selecting the rule for atomic invariants, such as $\text{DI}_=$, Lie, P-c, etc.

The proof strategy would also need to be instantiated with the proof rules for invariant equations which would be used in inferences marked by ? in the proof tree skeleton. In fact, one may use any suitable proof rule (even $\text{DRI}_\wedge$) to close the left branch in DC once it is reduced to proving invariance.

Unlike with decision procedures, such as $\text{DRI}_\wedge$, knowledge about the system is often crucial for differential cuts to be effective; however, this knowledge can sometimes be used to construct proofs that are more computationally efficient. We identify an example (Example 71) of an invariant with 13 state variables which defeats currently available decision procedures and which is easily provable using differential cuts together with both $\text{DI}_=$ and Lie. Though very much an artificial problem, it demonstrates that structure in the problem can sometimes be exploited to yield efficient proofs using DC. This is especially useful for large systems with many variables where the structure of the problem is well-understood. Additionally, we see that a combination of proof rules $(\text{DI}_=, \text{Lie}, \text{DC})$ can be both helpful and efficient.

While differential cuts can serve to increase the deductive power of sufficient proof rules, there are invariant conjunctions of equalities for which applying DC on the conjuncts given in the problem will altogether fail to be fruitful. This is due to the fact that at least some of the conjuncts, when considered individually, are required to define continuous invariants for the system (which may not be the case even if the conjunction is invariant). For example, equilibrium points can be represented as sum-of-squares or a conjunction of equations,

$$(x_1 - a_1)^2 + \cdots + (x_n - a_n)^2 = 0 \quad \equiv \quad x_1 = a_1 \wedge \cdots \wedge x_n = a_n,$$

where $(a_1, \ldots, a_n) \in \mathbb{R}^n$ is an equilibrium for the system, i.e. $f(\vec{a}) = \vec{0}$. Geometrically, this corresponds to an intersection of real hyperplanes, none of which may be invariant under the flow of $\dot{\vec{x}} = f(\vec{x})$ when considered separately, whereas their intersection describes an invariant set.

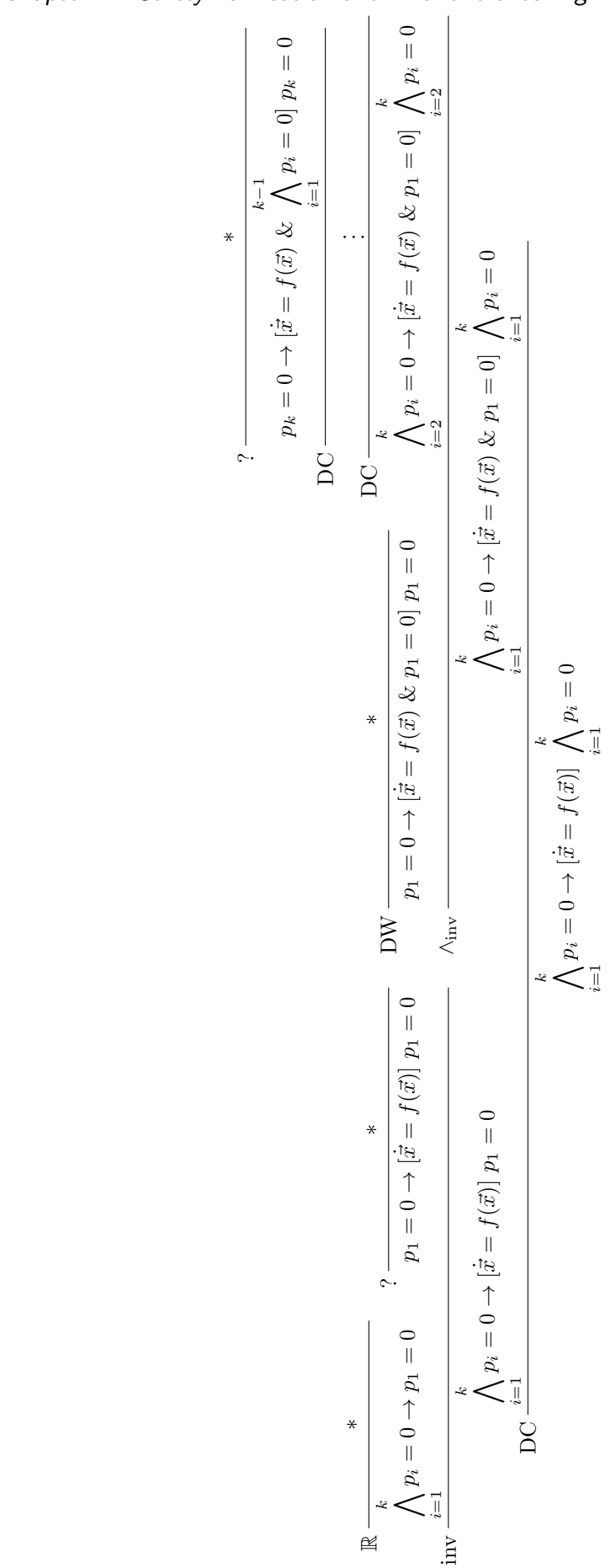


Figure 4.15: Proof strategy for proving invariance of real algebraic varieties using differential cuts (DC).

Example 71

Consider the system

$$
\begin{aligned}
\dot{x}_1 &= -292x_7(-1+x_6^2+x_7^2+x_8^2)^{145},\\
\dot{x}_2 &= -292x_8(-1+x_6^2+x_7^2+x_8^2)^{145},\\
\dot{x}_3 &= -42(2x_{10}+2x_{10}^3+2x_9)(-3+6x_{10}^2+x_{10}^4+2x_{10}x_9+2x_{10}^3x_9+x_9^2)^{41},\\
\dot{x}_4 &= -42(12x_{10}+4x_{10}^3+2x_9+6x_{10}^2x_9)(-3+6x_{10}^2+x_{10}^4+2x_{10}x_9+2x_{10}^3x_9+x_9^2)^{41},\\
\dot{x}_5 &= -2x_{13}(-1+x_{13}+x_{11}x_{13}),\\
\dot{x}_6 &= -2x_{12}(-1+x_{12}+x_{11}x_{12}),\\
\dot{x}_7 &= 26(-6x_1x_2^2+4x_1^3x_2^2+2x_1x_2^4)(1-3x_1^2x_2^2+x_1^4x_2^2+x_1^2x_2^4)^{25},\\
\dot{x}_8 &= 26(-6x_1^2x_2+2x_1^4x_2+4x_1^2x_2^3)(1-3x_1^2x_2^2+x_1^4x_2^2+x_1^2x_2^4)^{25},\\
\dot{x}_9 &= 14(4x_3^3x_4^2+2x_3x_4^4-6x_3x_4^2x_5^2)(x_3^4x_4^2+x_3^2x_4^4-3x_3^2x_4^2x_5^2+x_5^6)^{13},\\
\dot{x}_{10} &= 14(2x_3^4x_4+4x_3^2x_4^3-6x_3^2x_4x_5^2)(x_3^4x_4^2+x_3^2x_4^4-3x_3^2x_4^2x_5^2+x_5^6)^{13},\\
\dot{x}_{11} &= 14(-6x_3^2x_4^2x_5+6x_5^5)(x_3^4x_4^2+x_3^2x_4^4-3x_3^2x_4^2x_5^2+x_5^6)^{13},\\
\dot{x}_{12} &= 292x_6(-1+x_6^2+x_7^2+x_8^2)^{145},\\
\dot{x}_{13} &= -x_{13}.
\end{aligned}
$$

Suppose the invariant candidate is given by the following conjunction:

$$
\begin{aligned}
x_{13}=0 \quad \wedge \quad ((x_1^4x_2^2+x_1^2x_2^4-3x_1^2x_2^2+1)^{13})^2 \quad &+\\
((x_3^4x_4^2+x_3^2x_4^4-3x_3^2x_4^2x_5^2+x_5^6)^7)^2 \quad &+\\
((-1+x_6^2+x_7^2+x_8^2)^{73})^2 \quad &+\\
((-3+6x_{10}^2+x_{10}^4+2x_{10}x_9+2x_{10}^3x_9+x_9^2)^{21})^2 \quad &+\\
(x_{12}+x_{11}x_{12}-1)^2=0.&
\end{aligned}
$$

By using a differential cut to restrict the evolution domain to the invariant manifold $x_{13}=0$ (using the rule Lie), we obtain a system for which the sum-of-squares conjunct is a Hamiltonian and thus a first integral; its zero level set can be easily proved to be an invariant using the rule $\mathrm{DI}_{=}$. Naïvely attempting to use a decision procedure takes an unreasonable amount of time due to the high degrees involved, while the proof involving DC takes under a second for both branches, provided the right rules are selected to prove invariance of atoms. ■

4.4 Checking invariants with inequalities

This section will review proof methods for checking invariants described using inequalities. Some of the methods developed for this problem were inspired by established results on stability analysis from control theory. In what follows, we will briefly touch upon these for completeness of presentation.

4.4.1 Lyapunov-like functions (barrier certificates)

An equilibrium in a system of ODEs $\dot{\vec{x}} = f(\vec{x})$ is a point $\vec{x}_0 \in \mathbb{R}^n$ such that $f(\vec{x}_0) = \vec{0}$. Such an equilibrium is called *stable* if for all $\epsilon > 0$ there exists some $\delta > 0$ such that starting inside any δ-neighbourhood of $\vec{x}_0$, $U_\delta(\vec{x}_0) \subset \mathbb{R}^n$, the solution $\varphi_t(\vec{x}_0)$ remains inside the ϵ-neighbourhood $U_\epsilon(\vec{x}_0)$ for all time $t \geq 0$. Intuitively, one may think of stability as requiring those solutions that start "close enough" to the equilibrium to remain "near" at all future times. This type of stability is known as *stability in the sense of Lyapunov* (often shortened i.s.L.) and is weaker than the most commonly sought-after property of *asymptotic stability*, which additionally requires all solutions to converge to the equilibrium. A weaker notion known as *Lagrange stability* simply requires the solutions to remain bounded (see e.g. [Gyf63]; for a broad overview of stability theory see [BS70]).

First introduced in [Lya92], Lyapunov functions are commonly used to demonstrate stability (i.s.L.) of equilibria in systems of non-linear ODEs. Informally, a Lyapunov function $V : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable *positive-definite* function of the system's state, whose value can never increase along the solutions of the system.

Theorem 72 *(Lyapunov's direct method).* *Given a system $\dot{\vec{x}} = f(\vec{x})$, defined on $\mathbb{R}^n$, if one can find a $V \in C^1(\mathbb{R}^n, \mathbb{R})$ such that*

$$V(\vec{x}) > 0 \quad \forall\ \vec{x} \in \mathbb{R}^n \setminus \{0\},$$
$$\nabla V \cdot f(\vec{x}) = \frac{d}{dt} V(\varphi_t(\vec{x})) \leq 0 \quad \forall\ \vec{x} \in \mathbb{R}^n,$$

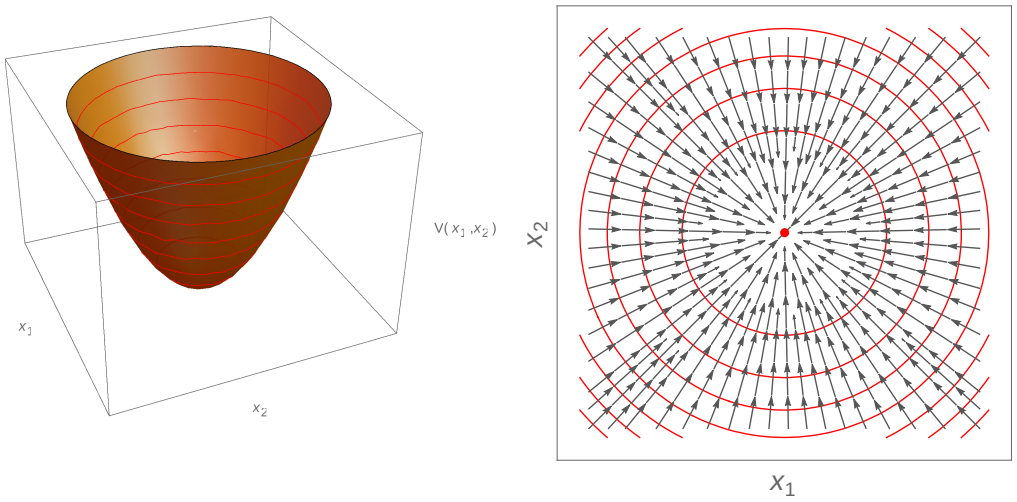
then one can conclude that the origin is stable (in the sense of Lyapunov).

Example 73 *Lyapunov function*

Consider the system

$$\dot{x}_1 = -x_1^3,$$
$$\dot{x}_2 = -x_2^3.$$

The equilibrium at the origin in this system is stable, which can be demonstrated using the Lyapunov function $V(\vec{x}) = x_1^2 + x_2^2$.



(a) Graph of the Lyapunov function V, giving an *abstract energy landscape*.

(b) Level sets of V bounding positively invariant regions in the system.

Figure 4.16: Lyapunov function $V(\vec{x}) = x_1^2 + x_2^2$ for the system $\dot{x}_1 = -x_1^3, \dot{x}_2 = -x_2^3$.

■

Lyapunov functions are of interest to safety verification because their sub-level sets are positively invariant. That is, if V is a Lyapunov function for $\dot{\vec{x}} = f(\vec{x})$, then $V \leq k$ defines a positively invariant set for every $k \geq 0$ [Bla99].

In [PJ04] Prajna and Jadbabaie introduced a methodology for safety verification using so-called *barrier certificates*, employing Lyapunov-like conditions to argue for safety, rather than stability.

Theorem 74 *(Barrier certificate [PJ04]).* *Given a system $\dot{\vec{x}} = f(\vec{x})$ & H under some evolution constraint $H \subseteq \mathbb{R}^n$, a set of initial states $\psi \subseteq H$ and a set of unsafe states $\neg\phi \subseteq H$, if one can find a continuously differentiable function*

$B : H \to \mathbb{R}$ *such that*

$$\begin{aligned} B(\vec{x}) &\leq 0 \quad \forall \vec{x} \in \psi, \\ B(\vec{x}) &> 0 \quad \forall \vec{x} \notin \phi, \\ \nabla B \cdot f(\vec{x}) &\leq 0 \quad \forall \vec{x} \in H, \end{aligned}$$

then the system is guaranteed to be safe.

Proof. See [Pra05, Proof of Proposition 2.2], or [PJ04, Proof of Proposition 1]. □

The problem of safety verification with barrier certificates is that of finding a function B that satisfies the conditions in the above theorem.

Example 75 *Prajna [Pra05]*
Consider a smooth continuous dynamical system characterised by

$$\begin{aligned} \dot{x_1} &= x_2 \\ \dot{x_2} &= -x_1 + \frac{1}{3}x_1^3 - x_2 \ . \end{aligned}$$

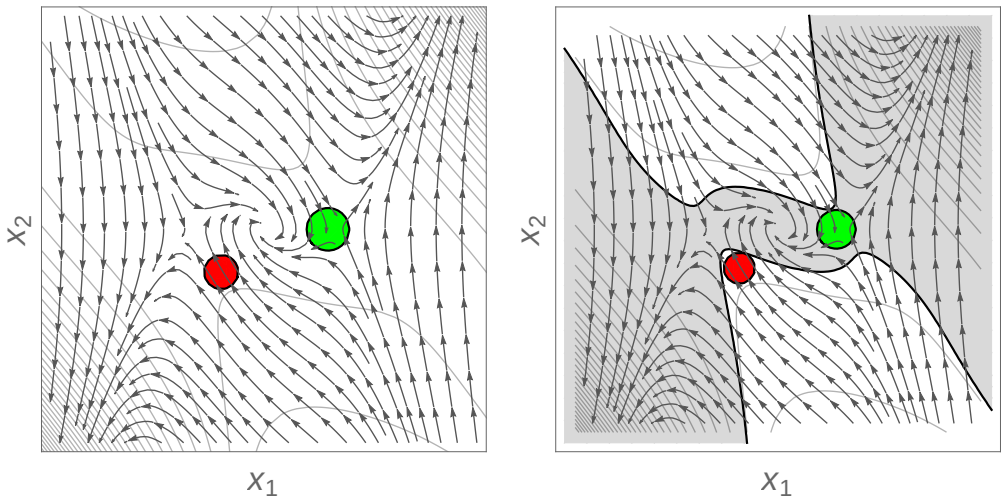
Figure 4.20a shows the system's phase portrait. The system is initialised at $x_0 \in (x_1 - 1.5)^2 + x_2^2 \leq 0.25 \subset \mathbb{R}^2$. We are further given a safety specification from which it follows that the system is unsafe if it transitions into the set of states where $(x_1 + 1)^2 + (x_2 + 1)^2 \leq 0.16$. These sets, ψ and $\neg\phi$, are shown in green and red, respectively.

■

The condition requiring the derivative of the barrier certificate function B to be non-positive everywhere within the evolution constraint H can in practice be rather restrictive, but also ensures that sum-of-squares optimisation techniques developed for Lyapunov functions [Par00] can be applied to automatically search for barrier certificates.

Alternatively, one may instead require the derivative of B to be strictly negative, but *only on the zero level set of* B *within* H. Such a B is known as a *strict* barrier certificate.

Theorem 76 *(Strict barrier certificate [PJ04, SPW12]).* *Given a safety verification problem for the continuous system* $\dot{\vec{x}} = f(\vec{x})$ & H *as before, if one*



(a) Level curves of B. (b) $B(x) \leq 0$.

Figure 4.17: Safety proof using barrier certificates, ψ shown in green, $\neg\phi$ in red and forward invariant sub-level set $B(\vec{x}) \leq 0$ in grey.

can find a continuously differentiable function $B : H \to \mathbb{R}$ *such that*

$$\begin{aligned} B(\vec{x}) &\leq 0 \quad \forall \vec{x} \in \psi, \\ B(\vec{x}) &> 0 \quad \forall \vec{x} \notin \phi, \\ \nabla B \cdot f(\vec{x}) &< 0 \quad \forall \vec{x} \in H \text{ s.t. } B(\vec{x}) = 0, \end{aligned}$$

then the system is guaranteed to be safe.

Proof. See [Pra05, Proof of Proposition 2.3]. □

Example 77 *Strict barrier certificate*

Consider the Wien bridge oscillator system from [Kha92, Exercise 1.23]:

$$\begin{aligned} \dot{x}_1 &= \frac{-g(x_2) - x_1 + x_2}{C_1 R_1}, \\ \dot{x}_2 &= -C_2 R_1 (-g(x_2) - x_1 + x_2) - \frac{x_2}{C_2 R_2} \end{aligned}$$

with $g(x) = \frac{333}{500}x^5 - \frac{439}{200}x^3 + \frac{1617}{500}x$, $H = \mathbb{R}^2$, and parameters $C_1 = C_2 = R_1 = R_2 = 1$, which results in the polynomial system

$$\begin{aligned} \dot{x}_1 &= -\frac{333}{500}x_2^5 + \frac{439}{200}x_2^3 - \frac{1117}{500}x_2 - x_1, \\ \dot{x}_2 &= \frac{333}{500}x_2^5 - \frac{439}{200}x_2^3 + \frac{617}{500}x_2 + x_1. \end{aligned}$$

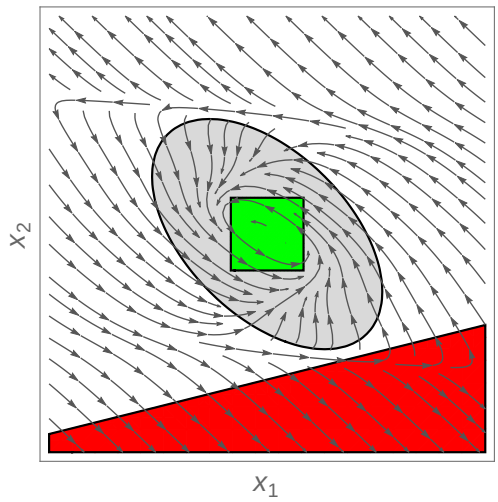


Figure 4.18: Safety proof using a strict barrier certificate B. The set $B \leq 0$ is shown in grey, possible choices for the initial states ψ in green and unsafe states $\neg\phi$ in red.

Using $B = x_1^2 + x_2 x_1 + x_2^2 - \frac{111}{59}$ and taking any initial and unsafe sets such that $\psi \to B \leq 0$ and $\neg\phi \to B > 0$, the function B acts as a strict barrier certificate which is sufficient to prove safety since we have $B(\vec{x}) = 0 \to \nabla B \cdot f(\vec{x}) < 0$.

Notice that B cannot act as a barrier certificate in the sense of Theorem 74 because $\nabla B \cdot f(\vec{x})$ can take positive values inside the sub-level set. ■

Theorem 78 *(Barrier certificate (non-convex type) [Pra05], Proposition 2.18).* *Given a safety verification problem for the continuous system $\dot{\vec{x}} = f(\vec{x})$ & H as before, but with $H \subseteq \mathbb{R}^n$* ***open****, if one can find a continuously differentiable function $B : H \to \mathbb{R}$ such that*

$$\begin{aligned} B(\vec{x}) &\leq 0 \quad \forall \vec{x} \in \psi, \\ B(\vec{x}) &> 0 \quad \forall \vec{x} \notin \phi, \\ \nabla B \cdot f(\vec{x}) &\leq 0 \quad \forall \vec{x} \in H \text{ s.t. } B(\vec{x}) = 0, \\ \nabla B(\vec{x}) &\neq \vec{0} \quad \forall \vec{x} \in H \text{ s.t. } B(\vec{x}) = 0, \end{aligned}$$

then the system is guaranteed to be safe.

Proof. See [Pra05, Section 2.5, pages 36–38]. In fact, the main principle at work here can be seen to be little more than an application of Nagumo's theorem (à la Bony [Bon69]; see [Red72],[Wal98, Chapter III, pages 117–119]). □

Example 79 *Barrier certificate (non-convex type)*
Let the continuous system [2] be given by

$$\dot{x}_1 = x_2 - x_1^7 \left(x_1^4 + 2\left(x_2^2 - 5\right)\right),$$
$$\dot{x}_2 = -3\left(x_1^4 + 2\left(x_2^2 - 5\right)\right) x_2^5 - x_1^3,$$

and evolving under no constraints, i.e. $H = \mathbb{R}^2$.

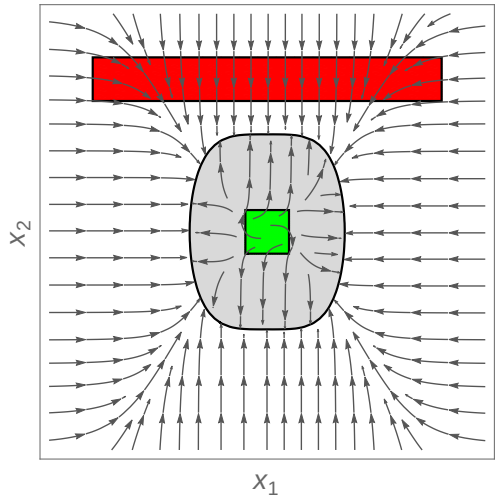


Figure 4.19: Safety proof using a non-convex barrier certificate B (based on Nagumo's theorem). The set $B \leq 0$ is shown in grey, possible choices for the initial states ψ in green and unsafe states $\neg\phi$ in red.

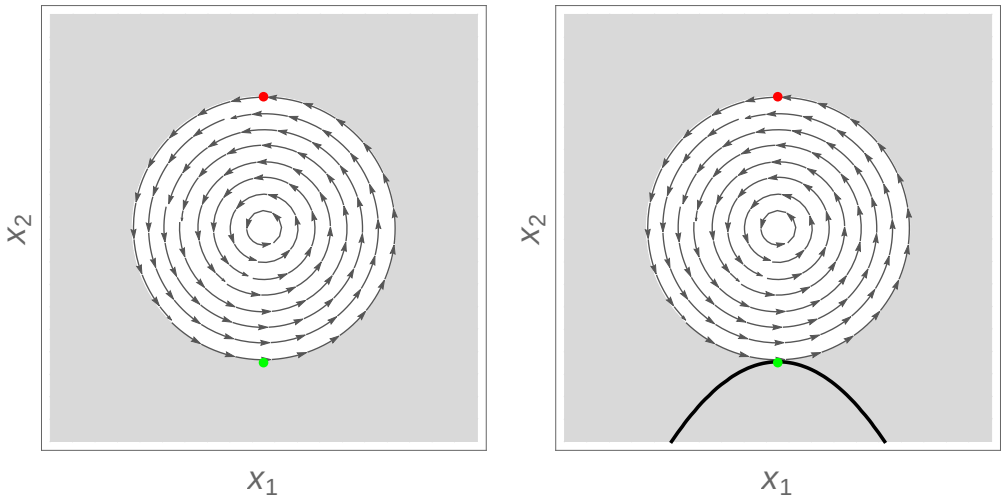
Using $B = x_1^4 + 2x_2^2 - 10$, one may show that the set $B \leq 0$ is positively invariant under the flow. Taking initial and unsafe sets such that $\psi \to B \leq 0$ and $\neg\phi \to B > 0$ are valid formulas, B can act as a non-convex barrier certificate, proving system safety by Theorem 78 because we have $B(\vec{x}) = 0 \to \nabla B \cdot f(\vec{x}) \leq 0$ and the zero level set of B is smooth, i.e. $B = 0 \to \nabla B \neq \vec{0}$ holds. Note that B is neither a barrier certificate, nor a strict barrier certificate. ■

Example 80 *Counterexample*
Consider the harmonic system

$$\dot{x}_1 = -x_2,$$
$$\dot{x}_2 = x_1$$

[2] System of ODEs taken from
`http://ocw.mit.edu/courses/aeronautics-and-astronautics/16-30-feedback-control-systems-fall-2010/lecture-notes/MIT16_30F10_lec22.pdf`

evolving inside the closed unit disc, i.e. $H \equiv x_1^2 + x_2^2 \leq 1$.



(a) Harmonic system confined to a closed unit disc $H \equiv x_1^2 + x_2^1 \leq 1$ ($\neg H$ in grey). (b) The curve $x_1^2 + x_2 + 1 = 0$ intersecting H precisely at $(0, -1)$.

Figure 4.20: The importance of open evolution constraints when using Nagumo-based conditions. Initial state $\psi \equiv x_1 = 0 \wedge x_2 = -1$ (in green) and an unsafe state $\neg\phi \equiv x_1 = 0 \wedge x_2 = 1$ (in red).

Figure 4.20 shows the system's phase portrait inside the constraint. Consider the set of initial states ψ to be the point $\vec{x}_0 = (0, -1) \in H$. This point is clearly not a fixed point of the system. If one is to consider the polynomial $x_1^2 + x_2 + 1$, one sees that the zero level set $x_1^2 + x_2 + 1 = 0$ intersects with H at precisely the point $\vec{x}_0$, i.e. $H \wedge x_1^2 + x_2 + 1 = 0 \equiv_{\mathbb{R}} \psi$. Now, taking $\neg\phi \equiv x_1 = 0 \wedge x_2 = 1$ to be the set of unsafe states, the system initialised in ψ is clearly unsafe because both ψ and $\neg\phi$ can be shown to lie on the same orbit characterised by the unit circle $x_1^2 + x_2^2 = 1$.

If we take $B = x_1^2 + x_2 + 1$ we see that it satisfies all the conditions from Theorem 78. In particular, one has $\nabla B \cdot f(\vec{x}) = 0$ for all $\vec{x} \in H$ s.t. $B(x) = 0$ (because the vector $f(\vec{x}_0)$ is tangent to the curve $B = 0$ at the only point where it touches H). In this example the theorem does *not* apply because the crucial assumption about H being an open set is not met. ■

One may formalise the method of barrier certificates to arrive at rules of inference for proving invariance of sets described by inequalities, which we now give.

Proposition 81. *The proof rules* BC*,* BCS *and* BCN *are sound.*

$$(\text{BC})\frac{H \vdash \nabla p \cdot f(\vec{x}) \leq 0}{\vdash p \leq 0 \rightarrow [\dot{\vec{x}} = f(\vec{x}) \ \& \ H]\ p \leq 0}$$

$$(\text{BCS})\frac{H \vdash p = 0 \rightarrow \nabla p \cdot f(\vec{x}) < 0}{\vdash p \leq 0 \rightarrow [\dot{\vec{x}} = f(\vec{x}) \ \& \ H]\ p \leq 0}$$

$$(\text{BCN})\frac{\vdash p = 0 \rightarrow (\nabla p \cdot f(\vec{x}) \leq 0 \wedge \nabla p \neq \vec{0})}{\vdash p \leq 0 \rightarrow [\dot{\vec{x}} = f(\vec{x}) \ \& \ H]\ p \leq 0}$$

Proof. Corollary to theorems 74, 76 and 78, by taking $\psi \equiv p \leq 0$, $\neg\phi \equiv p > 0$ and $B = p$. In the case of BCN, H in the conclusion may be safely dropped since it is not assumed in the premise, i.e. H is assumed to be $\mathbb{R}^n$, but was added here for uniformity of presentation. □

4.4.2 Invariant inequalities with encoded boolean structure

Similar to square-free reduction for invariant polynomial equations, one may sometimes remove roots of multiplicities greater than 1 from polynomial inequalities $p \leq 0$, thereby simplifying their description and removing singularities on their boundary. To do this, we will require some definitions.

Definition 82 *(Square-free decomposition [Yun76]).* *Given a polynomial $p \in \mathbb{Z}[\vec{x}]$, the square-free decomposition is given by*

$$(p_1, \ldots, p_n) \ \textit{s.t.} \ \prod_{i=1}^{n} p_i^i = p,$$

where all p_i are square-free and relatively prime.

Note that while superficially similar to square-free reduction, the square-free decomposition is quite different. To see this, note that the exponent in the product matches the index. Thus, the order in a square-free decomposition encodes the exponent to which the factor p_i is raised in the original polynomial p, i.e., the factors raised to odd powers will have odd index in the decomposition; respectively for even exponents.

Definition 83 *(Parity decomposition [DS95]).* *Given a polynomial $p \in \mathbb{Z}[\vec{x}]$ with square-free decomposition $(p_1, \ldots, p_n)$, the parity decomposition is given by*

$$\left(\prod_{odd\ i} p_i, \prod_{even\ i} p_i \right).$$

Proposition 84 *([DS95]).* *Let $p \in \mathbb{Z}[\vec{x}]$ and let (p_o, p_e) be the parity decomposition of p. Then the following real arithmetic equivalences hold:*

1. $p = 0 \equiv_{\mathbb{R}} \mathrm{SF}(p) = 0$,
2. $p \neq 0 \equiv_{\mathbb{R}} \mathrm{SF}(p) \neq 0$,
3. $p > 0 \equiv_{\mathbb{R}} p_o p_e^2 > 0 \equiv_{\mathbb{R}} p_o > 0 \wedge p_e \neq 0$,
4. $p \geq 0 \equiv_{\mathbb{R}} p_o p_e^2 \geq 0 \equiv_{\mathbb{R}} p_o \geq 0 \vee p_e = 0$,
5. $p < 0 \equiv_{\mathbb{R}} p_o p_e^2 < 0 \equiv_{\mathbb{R}} p_o < 0 \wedge p_e \neq 0$,
6. $p \leq 0 \equiv_{\mathbb{R}} p_o p_e^2 \leq 0 \equiv_{\mathbb{R}} p_o \leq 0 \vee p_e = 0$.

The resulting (rightmost) equivalent formulas are guaranteed to only feature square-free polynomials.

The following example demonstrates the use of parity decomposition in the case where the invariant candidate is of the form $p \leq 0$ and neither p_o nor p_e is trivial. With an extra reasoning step, one may reduce the invariant checking problem to two sub-problems involving single atomic relations involving only square-free polynomials.

Example 85 *Invariant defined by polynomial inequality*
Let us consider a system with an unstable limit cycle around a stable origin:

$$\dot{x}_1 = -x_1 - x_2 + x_1 x_2^2 + x_1^3,$$
$$\dot{x}_2 = x_1 - x_2 + x_1^2 x_2 + x_2^3.$$

Suppose we wanted to show that the set of states satisfying the following inequality is positively invariant:

$$(x_1^2 + x_2^2 - 1)^2 (x_1^2 + x_2^2 - \frac{1}{2})^3 \leq 0.$$

Let us refer to the above set as $p \leq 0$. As can be seen from the phase portrait in Figure 4.21, the set $p \leq 0$ is indeed positively invariant under the flow; however, p is *not* a barrier certificate that can be used to demonstrate invariance of $p \leq 0$ using the rules BC, BCS or BCN.

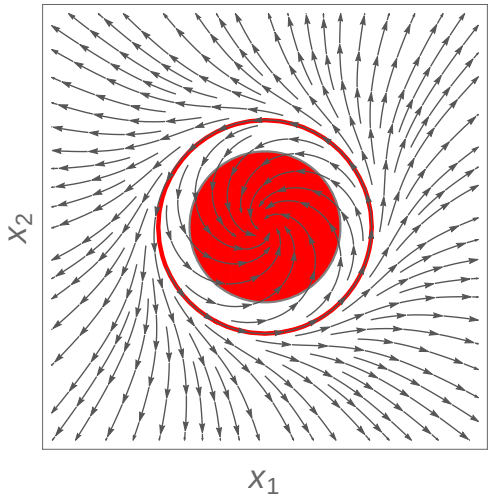


Figure 4.21: Positively invariant set given by $p \leq 0$ (in red).

Furthermore, p is not square-free, and $p \leq 0$ has the following parity normal:

$$\text{PNF}[(x_1^2 + x_2^2 - 1)^2(x_1^2 + x_2^2 - \frac{1}{2})^3 \leq 0] \equiv (x_1^2 + x_2^2 - \frac{1}{2}) \leq 0 \vee (x_1^2 + x_2^2 - 1) = 0.$$

What is perhaps remarkable is that the problem can be solved by working with the output of $\text{PNF}[p \leq 0]$, which yields two invariant checking sub-problems, both of which we can solve. One candidate is a non-strict inequality $x_1^2 + x_2^2 - \frac{1}{2} \leq 0$, whose defining polynomial can be used as a strict barrier certificate and checked using the rule BCS, the other is a polynomial equality $x_1^2 + x_2^2 - 1 = 0$ defining a smooth invariant curve, which we can also handle (using e.g. the proof rule Lie).

By performing the above steps one proves that both disjuncts are positively invariant under the flow, and hence their disjunction is also positively invariant, concluding the proof that $p \leq 0$ describes a positively invariant set. A formal proof of this property within a proof calculus would need to implement a barrier certificate-based inference rule such as BCS (or BCN), some appropriate rule for equational invariants, such as e.g. Lie, P-c or DRI, as well as the following:

$$(\text{inv}_\vee)\ \frac{\vdash S_1 \to [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ S_1 \quad \vdash S_2 \to [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ S_2}{\vdash S_1 \vee S_2 \to [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ S_1 \vee S_2}.$$

■

4.5 Semi-algebraic invariants

In this section we turn to the problem of semi-algebraic invariant checking.

4.5.1 Differential invariants

Differential induction with differential invariants (henceforth DI) was introduced in [Pla10a] and was shown to generalise the method of barrier certificates (non-strict barrier certificates were first introduced in [PJ04]) from sub-level sets of differentiable functions to formulas. In DI, F is a quantifier-free first-order formula in the theory of real arithmetic and D is the *derivation operator* [Pla12a, Definition 3.2], which is given as follows for terms:

$$\begin{aligned} D(r) &= 0 \quad \text{for numbers,} \\ D(x) &= \dot{x} \quad \text{for variables,} \\ D(a+b) &= D(a) + D(b), \\ D(a \cdot b) &= D(a) \cdot b + a \cdot D(b), \\ D\left(\frac{a}{b}\right) &= \frac{D(a) \cdot b - a \cdot D(b)}{b^2}, \end{aligned}$$

and in the following way for formulas:

$$\begin{aligned} D(a \leq b) &\equiv D(a) \leq D(b), \quad \text{accordingly for } \geq, >, <, =, \\ D(F \wedge G) &\equiv D(F) \wedge D(G), \\ D(F \vee G) &\equiv D(F) \wedge D(G), \ (\wedge \text{ here is important for soundness}). \end{aligned}$$

The formula $D(F)^{f(\vec{x})}_{\dot{\vec{x}}}$ is obtained by replacing each $\dot{x}_i$ in $D(F)$ with the corresponding right hand side in the system of differential equations, i.e. by $f_i(\vec{x})$.

Theorem 86 *([Pla10a]).* *The following inference rule is sound:*

$$(\text{DI})\frac{\vdash H \rightarrow D(F)^{f(\vec{x})}_{\dot{\vec{x}}}}{\vdash F \rightarrow [\dot{\vec{x}} = f(\vec{x}) \ \& \ H]\ F}.$$

Proof. See [Pla10a, Theorem 1]. □

Remark 87. Note that if one succeeds at finding an $F \equiv p \leq 0$, then a proof of invariance using the rule DI is equivalent to a proof of invariance using p as a barrier certificate. Differential invariants thus include barrier certificates as a special case [Pla10a].

Example 88

Consider the system

$$\dot{x}_1 = x_2^2,$$
$$\dot{x}_2 = 2,$$

and let $S \equiv x_2 \geq 0 \wedge x_1 \geq 0$.

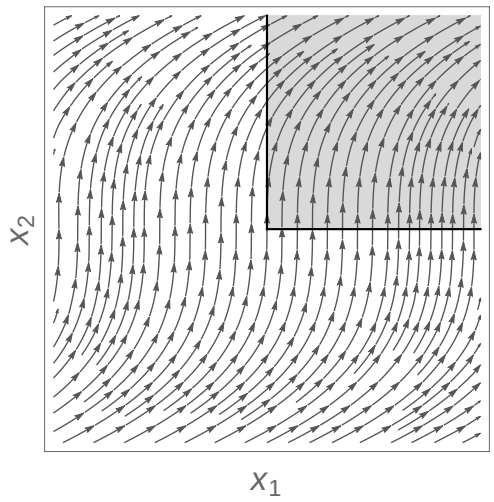


Figure 4.22: Phase portrait and a differential invariant S (in grey).

Using DI one may easily prove that S is a continuous invariant. Since $D(S) = \dot{x}_2 \geq 0 \wedge \dot{x}_1 \geq 0$ and therefore $D(S)^{f(\vec{x})}_{\dot{\vec{x}}} = 2 \geq 0 \wedge x_2^2 \geq 0$, which is equivalent to True in the theory of real arithmetic, the premise $\vdash H \rightarrow D(S)^{f(\vec{x})}_{\dot{\vec{x}}}$ of the rule DI is satisfied. ■

In practice, although differential invariants allow one to work with sets that are expressed using formulas with boolean operators, the conditions are very conservative (because they are required to hold everywhere in the state space, rather than only on the boundary of the set defined by F) and will often fail to hold even for seemingly simple continuous invariants.

4.5.2 Nagumo-like conditions for closed semi-algebraic sets

Nagumo's theorem (see Theorem 29) gives a necessary and sufficient condition for positive invariance of *closed* sets; however, one needs to be careful when applying this result to sets defined by formulas with logical connectives.

To appreciate this problem, we will require some basic facts about the properties of the contingent cone, i.e. the set of all sub-tangential vectors to a set S at a given point $\vec{x}$, denoted by $K_{\vec{x}}(S)$. The following lemma is useful when working with atomic formulas describing sub-level sets of differentiable functions.

Lemma 89 *([Pra05], Lemma 2.20).* *Let $S \equiv \{\vec{x} \in \mathbb{R}^n \mid B(\vec{x}) \leq 0\}$ where B is some continuously differentiable real valued function. Then $K_{\vec{x}}(S) = \mathbb{R}^n$ for all $\vec{x} \in \operatorname{int}(S)$ and $K_{\vec{x}}(S) = \{\vec{v} \in \mathbb{R}^n \mid \nabla B(\vec{x}) \cdot \vec{v} \leq 0\}$, for any $\vec{x}$ such that $B(\vec{x}) = 0$, provided that $\nabla B(\vec{x}) \neq \vec{0}$.*

In applications it is often tempting to apply the check for vector membership in the contingent cone *element-wise* to sets defining the atomic sub-formulas of a larger set, but in certain degenerate cases this leads to incorrect conclusions. Below we review the closure properties of the contingent cone.

Proposition 90. *Let $S_1, S_2 \subseteq \mathbb{R}^n$, then for all $\vec{x} \in S$ we have*

$$K_{\vec{x}}(S_1) \cup K_{\vec{x}}(S_2) \subseteq K_{\vec{x}}(S_1 \cup S_2).$$

Proof. Since $\operatorname{dist}(S, \cdot) \geq 0$ and $S_1 \subseteq S_1 \cup S_2$, we have

$$\begin{aligned} 0 \leq \quad & \inf_{\vec{x} \in S_1 \cup S_2} \|\vec{x} - \vec{x}_0\| \leq \quad \inf_{\vec{x} \in S_1} \|\vec{x} - \vec{x}_0\| \quad \text{for any } \vec{x}_0, \text{ and} \\ 0 \leq \quad & \operatorname{dist}(S_1 \cup S_2, \vec{x}_0) \leq \quad \operatorname{dist}(S_1, \vec{x}_0) \quad \text{by definition.} \end{aligned}$$

Substituting $\vec{x}_0 + t\vec{v}$ for $\vec{x}_0$ and dividing by $t > 0$ we get

$$\begin{aligned} 0 \leq \quad & \frac{\operatorname{dist}(S_1 \cup S_2, \vec{x}_0 + t\vec{v})}{t} \leq \quad \frac{\operatorname{dist}(S_1, \vec{x}_0 + t\vec{v})}{t} \quad \text{and by assumption} \\ 0 \leq \quad & \liminf_{t \to 0^+} \frac{\operatorname{dist}(S_1 \cup S_2, \vec{x}_0 + t\vec{v})}{t} \leq \quad \liminf_{t \to 0^+} \frac{\operatorname{dist}(S_1, \vec{x}_0 + t\vec{v})}{t} = 0. \end{aligned}$$

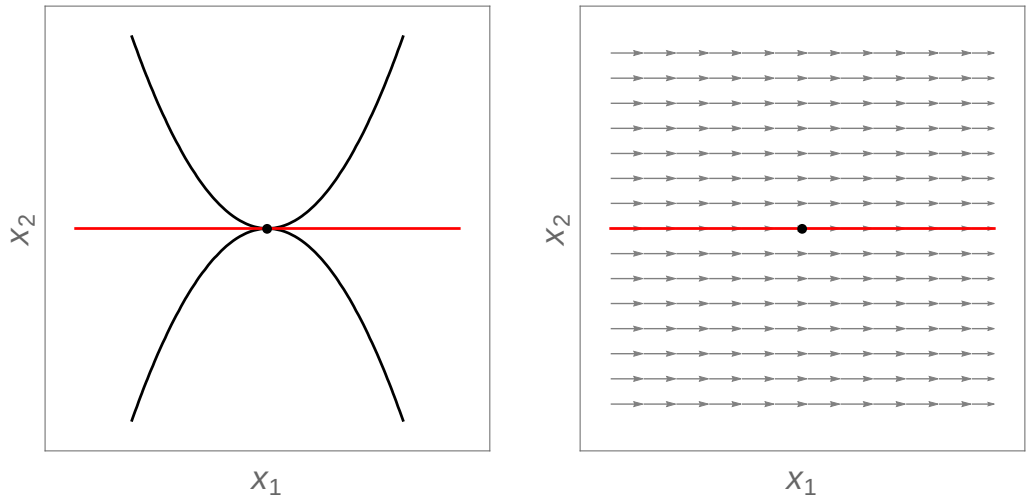
from which it follows that if $\vec{v}$ is sub-tangential to S_1 at $\vec{x}_0$, then it is also sub-tangential to $S_1 \cup S_2$. Thus, $K_{\vec{x}}(S_1) \subseteq K_{\vec{x}}(S_1 \cup S_2)$ for all $\vec{x} \in S_1$; by

the same argument one shows $K_{\vec{x}}(S_2) \subseteq K_{\vec{x}}(S_1 \cup S_2)$ for all $\vec{x} \in S_2$, from which one concludes that the inclusion $K_{\vec{x}}(S_1) \cup K_{\vec{x}}(S_2) \subseteq K_{\vec{x}}(S_1 \cup S_2)$ holds for all $\vec{x} \in S_1 \cup S_2$. □

Proposition 91. *Let $S_1, S_2 \subseteq \mathbb{R}^n$, then in general*

$$K_{\vec{x}}(S_1) \cap K_{\vec{x}}(S_2) \not\subseteq K_{\vec{x}}(S_1 \cap S_2).$$

Proof. Consider $S_1 \equiv \{\vec{x} \mid x_2 + x_1^2 = 0\}$ and $S_2 \equiv \{\vec{x} \mid x_2 - x_1^2 = 0\}$. The two sets intersect at $\vec{0} \in \mathbb{R}^2$. At the origin, the intersection of the contingent cones is given by the real line, i.e. $K_{\vec{0}}(S_1) \cap K_{\vec{0}}(S_2) = \{\vec{x} \mid x_2 = 0\}$, whereas the contingent cone to the intersection of the two sets is given by the zero vector, $K_{\vec{0}}(S_1 \cap S_2) = \{\vec{0}\}$. See Figure 4.23 for an illustration and [Wu10] for an overview of this problem.



(a) $x_2 + x_1^2 = 0 \wedge x_2 - x_1^2 = 0$ (b) $\dot{x}_1 = 1, \dot{x}_2 = 0$

Figure 4.23: Closure properties of the contingent cone at an intersection of two closed sets. The intersection of the contingent cones to the two sets is shown in red. The contingent cone to the intersection itself is $\{\vec{0}\}$.

□

In general, given a closed set S which is presented as a finite union of intersections of closed sets S_{ij}, i.e.

$$\bigcup_{i=1}^{k} \bigcap_{j=1}^{m(i)} S_{ij},$$

one would like to determine if $\vec{p}(\vec{x}) \in K_{\vec{x}}(S)$ by only performing checks $\vec{p}(\vec{x}) \in K_{\vec{x}}(S_{ij})$. If one has

$$\bigcup_{i=1}^{k} \bigcap_{j=1}^{m(i)} K_{\vec{x}}(S_{ij}) \subseteq K_{\vec{x}}(\bigcup_{i=1}^{k} \bigcap_{j=1}^{m(i)} S_{ij}). \tag{4.5}$$

for all $\vec{x}$ on the boundary of S, then Nagumo's criterion for vector field membership in the contingent cone for the whole set can be applied component-wise, i.e. the condition becomes

$$\forall \vec{x} \in \text{bdr} \left(\bigcup_{i=1}^{k} \bigcap_{j=1}^{m(i)} S_{ij} \right) . \quad \vec{p}(\vec{x}) \in \bigcup_{i=1}^{k} \bigcap_{j=1}^{m(i)} K_{\vec{x}}(S_{ij}).$$

It is possible to formulate inference rules based on Nagumo's theorem which allow one to prove positive invariance of a large class of closed semi-algebraic sets. This has previously been explored in [TT09], where a number of inference rules are presented for checking positive invariance of closed sets of the form $p \geq 0$. The lack of closure of the contingent cone at intersections of closed sets is the cause of unsoundness in the approach used in [TT09, p. 393,], which essentially requires each S_{ij} to be of the form $p_{ij} \geq 0$ and assumes the soundness-critical property (4.5). This deficiency can be fixed by e.g. requiring the matrix of partial derivatives of active components on the boundary of the conjunction to be full rank, i.e. $rk(\nabla p_1, \nabla p_2, \ldots, \nabla p_k) = k$ whenever the polynomials $p_1, p_2, \ldots, p_k$ evaluate to 0 on the boundary. A number of other possible *sufficient conditions* for removing this source of unsoundness have been studied in non-smooth analysis [Wu10] (see also *practical sets* in [BM08]). However, in practice, even ensuring the full-rank property for a matrix with polynomial entries is computationally expensive.

4.5.3 Non-smooth strict barrier certificates

A stronger criterion, which we term *non-smooth strict barrier certificate* may be seen as a generalisation of polynomial strict barrier certificates [PJP07] (formalised in the proof rule BCS) to closed semi-algebraic sets.

Proposition 92 *(Non-smooth strict barrier certificates).* *Given a continuous system* $\dot{\vec{x}} = f(\vec{x})$ *and a* closed *semi-algebraic set* $S \equiv \bigvee_{i=1}^{k} \bigwedge_{j=1}^{m(i)} p_{ij} \leq 0$, *where*

$p_{ij} \in \mathbb{R}[x_1, \ldots, x_n]$, *observe that* S *may be equivalently described by a sub-level set of a continuous function, i.e.*

$$S \equiv \bigvee_{i=1}^{k} \bigwedge_{j=1}^{m(i)} p_{ij} \leq 0 \equiv \min_{i=1,\ldots,k} \max_{j=1,\ldots,m(i)} p_{ij} \leq 0.$$

Let us inductively define $\mathfrak{L}_f(\max(p)) < 0 \equiv \mathfrak{L}_f(p) < 0$ *whenever* p *is a polynomial and*

$$\begin{aligned}
&\mathfrak{L}_f(\max(p_1, p_2, \ldots, p_m)) < 0 \equiv \\
&\quad \big(p_1 > \max(p_2, \ldots, p_m) \to \mathfrak{L}_f(p_1) < 0 \\
&\quad \wedge\, p_1 < \max(p_2, \ldots, p_m) \to \mathfrak{L}_f(\max(p_2, \ldots, p_m)) < 0 \\
&\quad \wedge\, p_1 = \max(p_2, \ldots, p_m) \to \mathfrak{L}_f(p_1) < 0 \wedge \mathfrak{L}_f(\max(p_2, \ldots, p_m)) < 0 \big).
\end{aligned}$$

For the min *function, we likewise define* $\mathfrak{L}_f(\min(g)) < 0 \equiv \mathfrak{L}_f(g) < 0$, *where* g *is either a polynomial or a* max *function with polynomial arguments. Otherwise, let*

$$\begin{aligned}
&\mathfrak{L}_f(\min(g_1, g_2, \ldots, g_m)) < 0 \equiv \\
&\quad \big(g_1 < \min(g_2, \ldots, g_m) \to \mathfrak{L}_f(g_1) < 0 \\
&\quad \wedge\, g_1 > \min(g_2, \ldots, g_m) \to \mathfrak{L}_f(\min(g_2, \ldots, g_m)) < 0 \\
&\quad \wedge\, g_1 = \min(g_2, \ldots, g_m) \to \mathfrak{L}_f(g_1) < 0 \vee \mathfrak{L}_f(\min(g_2, \ldots, g_m)) < 0 \big).
\end{aligned}$$

With these definitions, the following proof rule is sound:

$$\text{(NSBCS)} \frac{H \vdash \min_{i=1,\ldots,k} \max_{j=1,\ldots,m(i)} p_{ij} = 0 \to \mathfrak{L}_f \left(\min_{i=1,\ldots,k} \max_{j=1,\ldots,m(i)} p_{ij} \right) < 0}{S \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\; S}.$$

Proof. Consider an arbitrary point $\vec{x}_0 \in H$ such that

$$\min_{i=1,\ldots,k} \max_{j=1,\ldots,m(i)} p_{ij} \Big|_{\vec{x}_0} = 0,$$

then it is necessarily the case that for those *active* max arguments with indices i_* in the largest $I_* \subseteq \{1, \ldots, k\}$ such that

$$\max_{j=1,\ldots,m(i_*)} p_{i_* j} \Big|_{\vec{x}_0} = 0$$

for all $i_* \in I_*$, the condition

$$\mathfrak{L}_f \left(\max_{j=1,\ldots,m(i_*)} p_{i_* j} \right) \Big|_{\vec{x}_0} < 0$$

needs to hold for at least some $i_* \in I_*$. Without loss of generality, assume that at $\vec{x}_0$ there is one such i_*. The condition guarantees that for all active polynomial arguments of the max function, their Lie derivative is strictly negative at $\vec{x}_0$. Since Lie derivatives of polynomials under polynomial vector fields are also polynomial functions (and thus continuous), there exists an open neighbourhood around $\vec{x}_0$ inside which $\mathfrak{L}_f(p_{i_*j}) < 0$ is true for all $j \in \{1, \ldots, m(i_*)\}$. Thus, if the system is initialized at $\vec{x}_0$, it is guaranteed to enter the region where

$$\max_{j=1,\ldots,m(i_*)} p_{i_*j} < 0$$

and remain there for some non-empty time interval $(0, \epsilon)$, where $\epsilon > 0$, by following the solution $\varphi_t(\cdot)$, which implies that

$$\min_{i=1,\ldots,k} \max_{j=1,\ldots,m(i)} p_{ij}(\varphi_t(\vec{x}_0)) \leq 0$$

for all $t \in [0, \epsilon]$. The result follows by choosing a sufficiently small ϵ that works for all $\vec{x}_0$ on the boundary of S inside H. □

Example 93 *Non-smooth strict barrier certificate*
Consider the system

$$\dot{x}_1 = -\left(x_1^3 + x_2^2 x_1 - x_1 - x_2\right),$$
$$\dot{x}_2 = -\left(x_2^3 + x_1^2 x_2 - x_2 + x_1\right),$$

with $H = \mathbb{R}^2$ and let $S \equiv \left(x_1 - \frac{1}{3}\right)^2 + x_2^2 - 2 \leq 0 \wedge \left(x_1 + \frac{1}{3}\right)^2 + x_2^2 - 2 \leq 0$, which is a positively invariant set under the flow of the system.

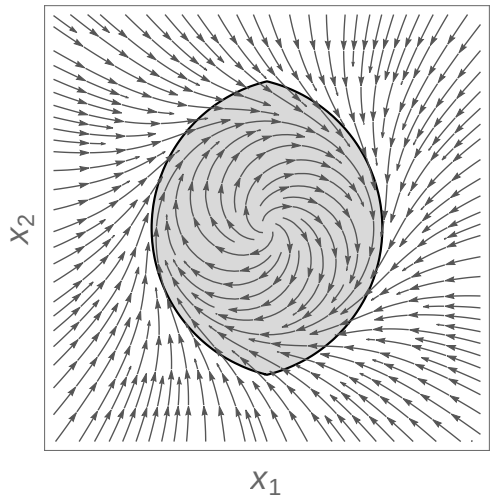


Figure 4.24: Phase portrait and a closed semi-algebraic positively invariant set S (in grey).

The invariance property cannot be proved using the rule DI, but is easily proved using the method of non-smooth strict barrier certificates with the rule NSBCS. ■

Remark 94. The rule NSBCS will fail to prove the invariance property in Example 88. Intuitively, this can be seen because at the origin the vector $f(\vec{0})$ does not point strictly into the interior of $S \equiv \max(-x_2, -x_1) \leq 0$, since $\mathfrak{L}_f(-x_1) = -x_2^2|_{\vec{0}} = 0$.

4.5.4 Decision procedure (Liu, Zhan & Zhao, 2011)

In [LZZ11] Liu, Zhan and Zhao have shown that it is *decidable* to check whether a given semi-algebraic set is positively invariant under the flow of a polynomial vector field. The conditions one is required to check are phrased in terms of set inclusion of semi-algebraic sets, which can be determined using a decision procedure for real arithmetic. The result builds on some of the ideas explored earlier by Taly and Tiwari in [TT09] and crucially depends on the property of solutions to differential equations with analytic right-hand sides being themselves analytic.

Lemma 95. *Let $p : \mathbb{R}^n \to \mathbb{R}$ be an analytic function and $\dot{\vec{x}} = f(\vec{x})$ be an analytic system of ODEs. If $\vec{x}_0 \in \mathbb{R}^n$ is such that $p(\vec{x}_0) = 0$, then one has three possibilities at $\vec{x}_0$:*

1. $\exists\ N > 0.\ \mathfrak{L}_f^N(p) < 0\ \wedge\ \bigwedge_{i=1}^{N-1} \mathfrak{L}_f^i(p) = 0$,
2. $\exists\ N > 0.\ \mathfrak{L}_f^N(p) > 0\ \wedge\ \bigwedge_{i=1}^{N-1} \mathfrak{L}_f^i(p) = 0$,
3. $\forall\ N > 0.\ \bigwedge_{i=1}^{N} \mathfrak{L}_f^i(p) = 0$.

In case 1, one has $p(\varphi_t(\vec{x}_0)) < 0$ for all $t \in (0, \epsilon)$ for some $\epsilon > 0$; case 2 is analogous, but with $p(\varphi_t(\vec{x}_0)) > 0$ for all $t \in (0, \epsilon)$. In case 3, one is guaranteed that $p(\varphi_t(\vec{x}_0)) = 0$ for all $t \in (0, \epsilon)$.

Proof. Since both p and the solution to the analytic ODE are analytic functions, the Taylor series expansion of $p(\varphi_t(\vec{x}_0))$ around $t = 0$ is given by

$$p(\vec{x}_0) + \sum_{i=1}^{\infty} \left(\frac{t^i}{i!} \cdot \left. \frac{d^i p}{dt^i} \right|_{\vec{x}_0} \right) = \sum_{i=1}^{\infty} \left(\frac{t^i}{i!} \cdot \mathfrak{L}_f^i(p)\big|_{\vec{x}_0} \right)$$

and *converges* on some non-empty open interval of t containing zero. Thus, the most significant term to become sign-definite will determine the sign of the entire sum on some sufficiently small interval. See [LZZ11, Proof of Proposition 9]. See also [TT09, Proof of Theorem 7], which employed very much the same ideas as [LZZ11]. □

Theorem 96 *(Liu, Zhan & Zhao [LZZ11]).* *Given a locally Lipschitz continuous system $\dot{\vec{x}} = f(\vec{x})$, and a set $S \subseteq \mathbb{R}^n$, let us define*

$$\mathrm{In}_f(S) \equiv \{\vec{x} \in \mathbb{R}^n \mid \exists\, \epsilon > 0.\, \forall\, t \in (0, \epsilon).\, \varphi_t(\vec{x}) \in S\},$$
$$\mathrm{In}_{(-f)}(S) \equiv \{\vec{x} \in \mathbb{R}^n \mid \exists\, \epsilon > 0.\, \forall\, t \in (0, \epsilon).\, \varphi_{-t}(\vec{x}) \in S\},$$

where $\varphi_t(\vec{x})$ is the solution to the initial value problem $(\dot{\vec{x}} = f(\vec{x}), \varphi_0(\vec{x}) = \vec{x})$ at time t. The set S is positively invariant under the flow of the system if and only if the inclusions $\mathrm{In}_{(-f)}(S) \subseteq S \subseteq \mathrm{In}_f(S)$ hold, which implies soundness of the following rule of inference:

$$(\mathrm{LZZ})\frac{\vdash \big(\mathrm{In}_{(-f)}(S) \to S\big) \wedge (S \to \mathrm{In}_f(S))}{S \to [\dot{\vec{x}} = f(\vec{x})]\ S}.$$

In fact, the premise and the conclusion in the above rule are equivalent.

Proof. Simple corollary to [LZZ11, Theorem 19]. □

The *decidability* of checking the conditions in the Proposition 96 hinges on the ability to construct *semi-algebraic sets* $\mathrm{In}_f(S)$ whenever S is semi-algebraic. In [LZZ11] Liu, Zhan and Zhao make the crucial observation that whenever p is a polynomial and $\dot{\vec{x}} = f(\vec{x})$ is a system of polynomial ODEs, then the Lie derivatives $\mathfrak{L}_f^i(p)$ up to any order i are also polynomials. Using the fact that the ring of multivariate polynomials with coefficients in some Noetherian ring is also Noetherian (by Hilbert's basis theorem), the set $\mathrm{In}_f(p \leq 0)$ can be characterised

by a *finite* disjunction [LZZ11] (illustrated in Figures 4.25 and 4.26):

$$\begin{aligned} p < 0 \quad & \vee \\ (p = 0 \wedge \mathfrak{L}_f(p) < 0) \quad & \vee \\ (p = 0 \wedge \mathfrak{L}_f(p) = 0 \wedge \mathfrak{L}_f^2(p) < 0) \quad & \vee \\ \vdots \quad & \\ (p = 0 \wedge \mathfrak{L}_f(p) = 0 \wedge \mathfrak{L}_f^2(p) = 0 \wedge \cdots \wedge \mathfrak{L}_f^{N-1}(p) < 0) \quad & \vee \\ (p = 0 \wedge \mathfrak{L}_f(p) = 0 \wedge \mathfrak{L}_f^2(p) = 0 \wedge \cdots \wedge \mathfrak{L}_f^{N-1}(p) = 0 \wedge \mathfrak{L}_f^N(p) \leq 0). & \end{aligned}$$

The ascending chain property of Noetherian rings guarantees that there is a finite positive integer N such that for all $N' \geq N$ we have the following ideal membership:

$$\mathfrak{L}_f^N(p) \in \langle p, \mathfrak{L}_f(p), \ldots, \mathfrak{L}_f^{N-1}(p) \rangle,$$

The integer N may be found by e.g. using Gröbner bases to successively check for ideal membership of $\mathfrak{L}_f^N(p)$ in the ideal generated by the Lie derivatives of orders lower than N for $N = 1, 2, 3, \ldots$ until the ideal saturates (as with DRI). Once N is found, if the formula

$$(p = 0 \wedge \mathfrak{L}_f p = 0 \wedge \mathfrak{L}_f^2 p = 0 \wedge \cdots \wedge \mathfrak{L}_f^{N-1} p = 0 \wedge \mathfrak{L}_f^N p = 0)$$

holds, then for any $N' \geq N$ we have

$$(p = 0 \wedge \mathfrak{L}_f p = 0 \wedge \mathfrak{L}_f^2 p = 0 \wedge \cdots \wedge \mathfrak{L}_f^{N-1} p = 0 \wedge \mathfrak{L}_f^N p = 0 \wedge \cdots \wedge \mathfrak{L}_f^{N'} p = 0),$$

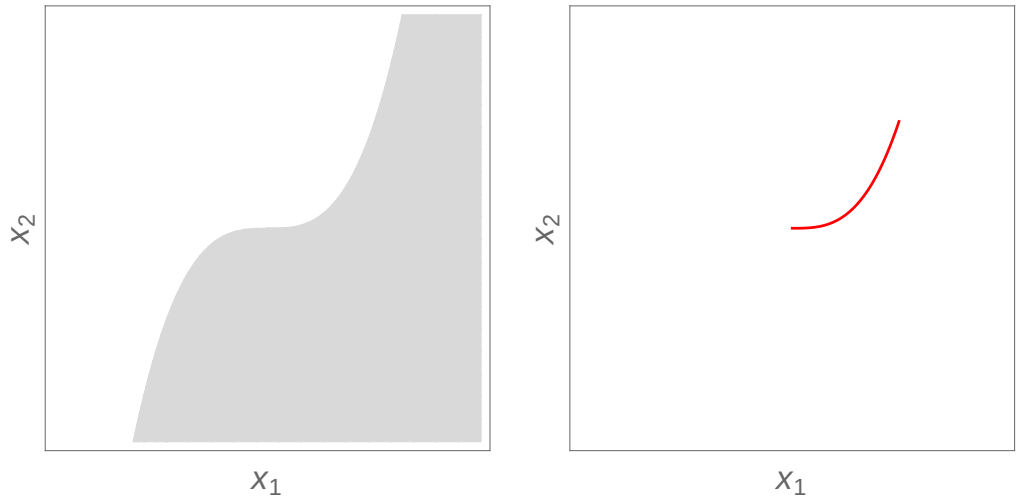
which removes the need to consider disjuncts with Lie derivatives of orders higher than N, as all the (infinitely many) formulas

$$(p = 0 \wedge \mathfrak{L}_f p = 0 \wedge \mathfrak{L}_f^2 p = 0 \wedge \cdots \wedge \mathfrak{L}_f^{N-1} p = 0 \wedge \mathfrak{L}_f^N p = 0 \wedge \cdots \wedge \mathfrak{L}_f^{N'} p < 0),$$

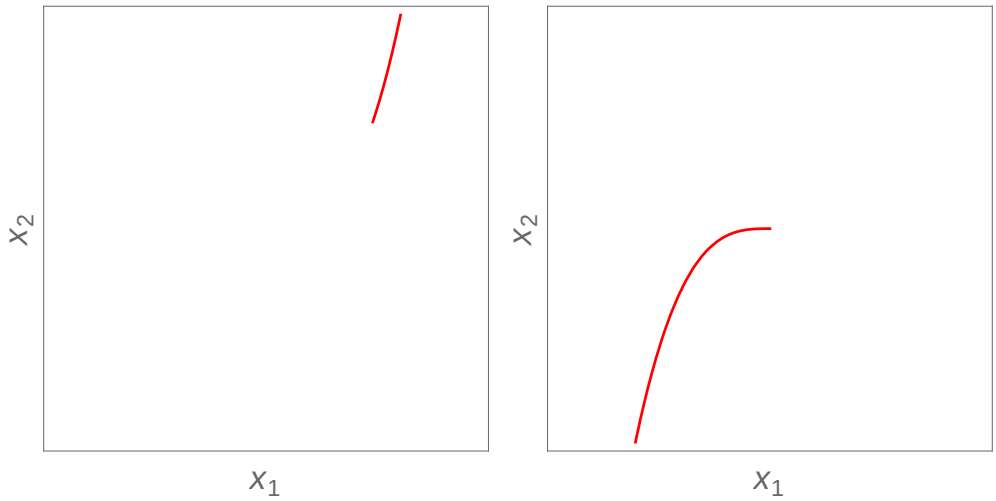
with $N' > N$ are guaranteed to be false.

Remark 97. The ascending chain property is crucial in making it possible to reason about sign conditions of *infinitely many* higher-order Lie derivatives by only considering a *finite* number of sign conditions. The same idea was independently pursued in [GP14a] to give a necessary and sufficient criterion for invariance of real algebraic sets under the flow of polynomial ODEs (summarised in the proof rule DRI; discussed earlier).

Thus, by computing N for a given polynomial p and a system $\dot{\vec{x}} = f(\vec{x})$, one may construct a *semi-algebraic* set $\mathrm{In}_f(p \leq 0)$.

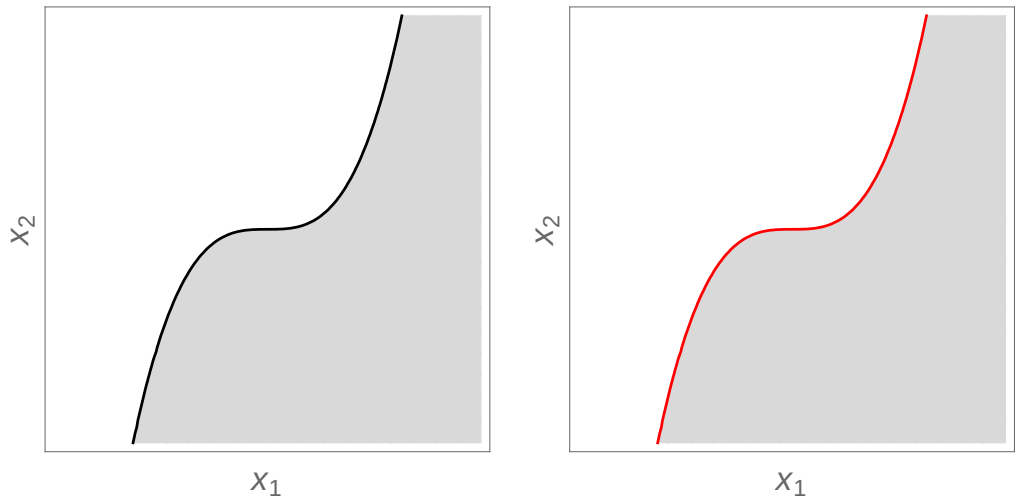


(a) $\mathfrak{L}_f^0(p) < 0$ (i.e. $p < 0$) (b) $p = 0 \wedge \mathfrak{L}_f(p) < 0$



(c) $p = \mathfrak{L}_f(p) = 0 \wedge \mathfrak{L}_f^2(p) < 0$ (d) $p = \mathfrak{L}_f(p) = \mathfrak{L}_f^2(p) = 0 \wedge \mathfrak{L}_f^3(p) \leq 0$

Figure 4.25: Sign conditions on Lie derivatives in the construction of $\mathrm{In}_f(p \leq 0)$ with $N = 3$.



(a) $p \leq 0$ (b) $\mathrm{In}_f(p \leq 0)$

Figure 4.26: Constructing $\mathrm{In}_f(p \leq 0)$ using higher-order Lie derivatives.

Likewise in the case of strict polynomial inequalities $p < 0$, the set $\mathrm{In}_f(p < 0)$ is semi-algebraic and is characterized by the following finite formula:

$$\begin{aligned} p < 0 \quad & \vee \\ (p = 0 \wedge \mathfrak{L}_f(p) < 0) \quad & \vee \\ (p = 0 \wedge \mathfrak{L}_f(p) = 0 \wedge \mathfrak{L}_f^2(p) < 0) \quad & \vee \\ \vdots \quad & \\ (p = 0 \wedge \mathfrak{L}_f(p) = 0 \wedge \mathfrak{L}_f^2(p) = 0 \wedge \cdots \wedge \mathfrak{L}_f^{N-1}(p) < 0) \quad & \vee \\ (p = 0 \wedge \mathfrak{L}_f(p) = 0 \wedge \mathfrak{L}_f^2(p) = 0 \wedge \cdots \wedge \mathfrak{L}_f^{N-1}(p) = 0 \wedge \mathfrak{L}_f^N(p) < 0). & \end{aligned}$$

To construct $\mathrm{In}_f(\cdot)$ for semi-algebraic sets with boolean structure, an important distribution property, proved in [LZZ11, Theorem 20], is required.

Theorem 98 *([LZZ11]).* *Given a polynomial system $\dot{\vec{x}} = f(\vec{x})$ and a semi-algebraic set $S \equiv \bigvee_{i=1}^{k} \bigwedge_{j=1}^{m(i)} p_{ij} \sim_{ij} 0$ where $\sim_{ij} \in \{<, \leq\}$, we have*

$$\mathrm{In}_f(S) \equiv \bigvee_{i=1}^{k} \bigwedge_{j=1}^{m(i)} \mathrm{In}_f(p_{ij} \sim_{ij} 0).$$

Proof. [LZZ11, Theorem 20]. □

Finally, let us note that $\mathrm{In}_{(-f)}(\cdot)$ can be constructed by computing $\mathrm{In}_f(\cdot)$ for the system with time-reversed dynamics $\dot{\vec{x}} = -f(\vec{x})$. This is possible because

$$\frac{d}{dt}\varphi_{-t}(\vec{x}) = -f(\varphi_t(\vec{x})),$$

and the solution to $\dot{\vec{x}} = -f(\vec{x})$ is given by $\varphi_{-t}(\vec{x})$, where $\varphi_t(\vec{x})$ is the solution to $\dot{\vec{x}} = f(\vec{x})$. The general result giving a decision procedure for checking semi-algebraic continuous invariants under the flow of polynomial ODEs with semi-algebraic constraints is summarised in the following theorem and follows from the fact that checking set inclusion is decidable for semi-algebraic sets.

Theorem 99 *(Liu, Zhan & Zhao [LZZ11]).* *Given a locally Lipschitz continuous system $\dot{\vec{x}} = f(\vec{x})$ & H and an initial set of states $\psi \subseteq \mathbb{R}^n$, the set S is a continuous invariant under the flow of the system if and only if $\psi \subseteq S$ and the following inclusions hold: $S \cap H \cap \mathrm{In}_f(H) \subseteq \mathrm{In}_f(S)$ and $\mathrm{In}_{(-f)}(S) \cap H \cap \mathrm{In}_{(-f)}(H) \subseteq S$. As a consequence, the premise and the conclusion in the following rule of inference are equivalent:*

$$(\mathrm{LZZ})\frac{\vdash (S \wedge H \wedge \mathrm{In}_f(H) \to \mathrm{In}_f(S)) \wedge \big(\mathrm{In}_{(-f)}(S) \wedge H \wedge \mathrm{In}_{(-f)}(H) \to S\big)}{S \to [\dot{\vec{x}} = f(\vec{x}) \ \&\ H]\ S}.$$

Proof. See [LZZ11, Theorem 19]. □

Example 100

Consider the system

$$\dot{x}_1 = -1,$$
$$\dot{x}_2 = -x_2$$

under constraint $H \equiv x_1 \geq -5$ and let $S \equiv -3 \leq x_2 \leq 3 \wedge -5 \leq x_1 \leq 0$.

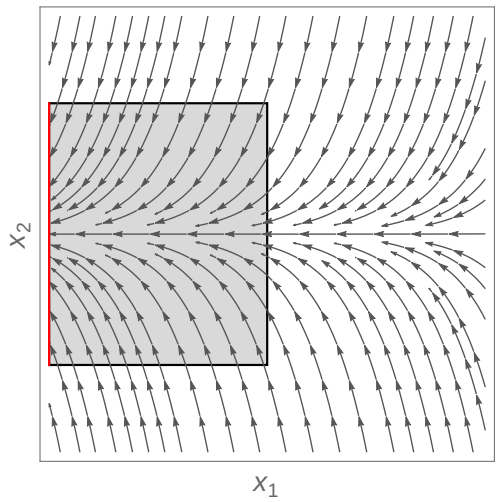


Figure 4.27: Phase portrait and a continuous invariant S (in grey). Region on the boundary of S and H where the conditions in premise of NSBCS are not satisfied is shown in red.

One cannot use DI or NSBCS to prove that S defines a continuous invariant for the system. This property can be proved easily using the decision procedure from [LZZ11]. Intuitively, NSBCS fails because at the boundary of S and H (highlighted in red in the figure), the vector field does not point strictly into S, however, as there can be no further motion in the system without violating the constraint H, the set S cannot be left from these states. ■

4.6 Summary

In this chapter we have formalised existing proof methods for continuous invariant checking as sound rules of inference and presented extensions (Lie°, Lie*, NSBCS). From the point of view of proof automation in a formal deductive system, it is perhaps desirable to have an array of sound proof rules for invariant checking; these rules should ideally lead to conditions that can be efficiently checked for certain classes of problems. The rules can further be combined into proof strategies to mechanise crucial parts of formal proofs that would otherwise be rather laborious to perform by hand. However, recent advances in algebraic and semi-algebraic invariant checking for polynomial ODEs lead to decision procedures, which theoretically removes the need to keep rules that are less

deductively powerful in the proof system. Furthermore, as we have seen in this chapter, many of the existing sufficient conditions are quite fragile and require great care in order to implement and use correctly. However, it is also true that sufficient conditions for invariant checking can sometimes afford a more efficient alternative to a decision procedure. Furthermore, when used inside a proof calculus implementing proof rules such as differential cut (DC), sufficient rules for invariant checking can be used to effectively exploit the structure in the problem.

The survey of continuous invariant checking methods given in this chapter is, to our knowledge, the most complete account of existing work on direct invariant checking. We have been explicit in highlighting soundness issues, technical subtleties and common pitfalls that users or developers wishing to use these methods to reason in a deductive framework might encounter.

Chapter 5

Invariant generation and discrete abstraction

Synopsis *In this chapter we will (**i**) introduce a method for constructing discrete abstractions of continuous systems in which transitions between the discrete states occur* if and only if *a corresponding continuous evolution is possible in the continuous system, thereby eliminating the problem of impossible transitions between discrete states in the abstraction, (**ii**) introduce algorithms for automatically generating (semi-algebraic) invariants for polynomial systems of ODEs under semi-algebraic evolution constraints by efficiently* extracting *reachable sets of discrete semi-algebraic abstractions, (**iii**) address the state explosion problem, associated with computing the discrete abstraction, by exploiting* differential cuts *and properties of invariant sets to reduce systems of ODEs to simpler sub-systems. (**iv**) We evaluate our invariant generation method on a collection of safety verification benchmarks featuring dynamical systems with interesting non-linear behaviour and (**vi**) conclude with a discussion of future directions and related work.*

5.1 Introduction

Sound approaches to safety verification rely on the existence of appropriate *invariants* that contain every state where the system may begin its evolution, and no unsafe states. Traditionally, there have been two popular methods for proving safety properties in continuous systems: one based on first *soundly abstracting* the continuous system and performing reachability analysis in the resulting discrete transition system, and a *deductive verification* approach that works with the invariants for the continuous system directly. Existing automatic procedures for invariant generation in deductive frameworks are able to produce invariants with simple boolean structure. Approaches based on abstraction, in computing reachable sets of discrete systems, (implicitly) handle invariants whose boolean structure is more intricate; their limitations currently stem from the conservative nature of the discrete models, whose transition behaviour is often a very coarse over-approximation of the evolution taking place in the continuous system.

Provided that one has the means of formally checking whether a given set of states defines an invariant that is sufficient to prove safety, one would ideally like to be able to *generate* appropriate invariants automatically. The safety property may then be verified inside a deductive prover without any manual steps on the part of the user.

A number of approaches have been proposed for generating invariants for continuous systems [PJ04, Tiw08b, GT08, SSM08, San10, LZZ11, ZZK13, GP14a, MMR11], which either put serious restrictions on the form of the invariant or rely on the user pre-defining a *parametric template* and then attempt to identify an instantiation of the parameters in the template that yields an invariant.

Remark 101. For further details about related work, see Section 5.8.

This chapter will pursue an alternative approach that automatically generates semi-algebraic continuous invariants from discrete semi-algebraic abstractions of continuous systems. Our rationale is that recent advances in semi-algebraic invariant checking for polynomial ODEs (due to Liu et al. [LZZ11]; see Chapter 4) allow deductive provers to work with arbitrary semi-algebraic invariants, yet few methods for invariant generation are able to synthesise interesting invariants with intricate boolean structure that one can find in reachable sets of discrete

abstractions. At the same time, existing discrete abstraction approaches (such as [TK02, Tiw08a] suffer from coarseness as well as unsoundness and do not take full advantage of the results on invariant checking in constructing the transition relation for the discrete transition system. In this chapter we will address both of these issues.

5.1.1 Invariant generation problem

Though the problem of semi-algebraic invariant checking has been solved for polynomial systems in [LZZ11], and therefore allows us to decide assertions of the form

$$I \rightarrow [\dot{\vec{x}} = f(\vec{x}) \ \& \ H] \ I \ ,$$

there remains the issue of semi-algebraic invariant *generation*, i.e. finding an appropriate continuous invariant I that is sufficient to prove the safety assertion

$$\psi \rightarrow [\dot{\vec{x}} = f(\vec{x}) \ \& \ H] \ \phi$$

using the proof rule (*Safety*) from Chapter 4, Section 4.1.

Let us begin by considering a rather unintelligent way of searching for such an invariant I by simply combining results about the decidability of first-order real arithmetic [Tar51] and semi-algebraic invariant checking [LZZ11]. We can formulate a "brute force" procedure for enumerating parametric semi-algebraic invariant templates that is *guaranteed* to find an appropriate semi-algebraic I, if one exists. We outline such a procedure in the proof to the following proposition.

Proposition 102. *The semi-algebraic continuous invariant generation problem for polynomial continuous systems is semi-decidable. More formally, given a safety assertion*

$$\psi \rightarrow [\dot{\vec{x}} = f(\vec{x}) \ \& \ H] \ \phi \ ,$$

where ψ, H and ϕ are quantifier-free formulas in the theory of real arithmetic, we can always compute a semi-algebraic continuous invariant I sufficient to prove system safety, if one exists.

Proof. Let $LM(\vec{x}, d)$ be the list of all monomials in the state variables up to degree $d > 0$ arranged with respect to some monomial ordering, e.g. $LM(\vec{x}, 1) = (1, x_1, x_2, \ldots, x_n)$. Now define $P(\vec{x}, \vec{\alpha}, d)$ to be a parametric polynomial formed by a linear combination of all the monomials in $LM(\vec{x}, d)$ with fresh formal symbols α_i as coefficients, e.g. $P(\vec{x}, \vec{\alpha}, 1) = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n$. For a given polynomial degree d, we can define *conjunctive semi-algebraic templates*, which are given by a conjunction of equalities and strict inequalities in the polynomial templates of maximum degree d, which share no formal parameters in their coefficients. We refer to conjunctive semi-algebraic templates as *cells*; each cell $C_i^d(k)$ (indexed by i) of maximum polynomial degree d is characterised by a conjunction of $2k$ parametric polynomial equalities and inequalities:

$$\begin{aligned} C_i^d(k) \equiv\ & P(\vec{x}, \vec{\alpha}_{(i,1)}, d) < 0 \wedge P(\vec{x}, \vec{\alpha}_{(i,2)}, d) = 0 \\ & \wedge\ P(\vec{x}, \vec{\alpha}_{(i,3)}, d) < 0 \wedge P(\vec{x}, \vec{\alpha}_{(i,4)}, d) = 0 \\ & \ \ \vdots \\ & \wedge\ P(\vec{x}, \vec{\alpha}_{(i,2k-1)}, d) < 0 \wedge P(\vec{x}, \vec{\alpha}_{(i,2k)}, d) = 0, \end{aligned}$$

where each list of formal parameters $\vec{\alpha}_{(i,j)}$ is unique. In general, a semi-algebraic template of maximum degree d is formed by a finite disjunction of cells of maximum degree d:

$$C_1^d(k) \vee C_2^d(k) \vee \cdots \vee C_m^d(k),$$

where the number of parametric polynomial equalities and inequalities defining each cell is $2k$. Increasing d, k or m thus results in a more general semi-algebraic template.

Let $(d, k, m) \in \mathbb{N}^3$ correspond to a semi-algebraic template with m cells each consisting of $2k$ parametric polynomial equalities and inequalities of maximum degree d. Semi-algebraic templates can therefore be put into one-to-one correspondence with a countable set and can thus be enumerated.

For each semi-algebraic template we can decide if there is an instantiation of formal parameters that yields a suitable continuous invariant [LZZ11, Theorem 29]. If a suitable semi-algebraic continuous invariant I exists, then it is composed of a finite disjunction of m_I conjunctive formulas, each with at most k_I polynomial equalities and inequalities of maximum polynomial degree that is bounded by d_I for all polynomials appearing in I. Hence, the semi-algebraic template

corresponding to the triple (d_I, k_I, m_I) will admit an instantiation of formal parameters that results in I. □

Checking parametric invariant templates suffers from the high computational cost associated with real quantifier elimination algorithms (e.g. CAD [Col75, CH91], which has doubly-exponential time complexity in the number of problem variables).

An invariant generation procedure that involves checking semi-algebraic templates with an ever-increasing number of fresh variables using a fundamentally infeasible procedure is sure to leave practically-minded users thoroughly unamused. Indeed, one cannot realistically hope to apply the brute-force invariant generation method even to very simple problems. A more practical way of searching for continuous invariants would need to be extremely cautious about introducing fresh variables (such as symbolic parameters) into the problem and would instead aim to work only with the *state variables*. In the next section we will begin to develop the foundations for a more scalable invariant generation method using discrete abstractions of continuous systems.

5.2 Discrete abstraction of continuous systems

Discrete abstraction is concerned with over-approximating (in a certain sense) a given continuous system by a finite discrete transition system. Such a transformation makes it possible to perform reachability analysis and thus verify safety properties in the simpler discrete model. The approach works by ensuring that the set of behaviours of the discrete (abstract) system *over-approximates* the set of behaviours of the continuous (concrete) system; this is known as *sound abstraction*. If the discrete abstraction is sound, then any violation of the safety property in the continuous system is necessarily reproduced by the abstract discrete transition system. Conversely, an abstraction is *complete* (with respect to the safety property) when any violation of the safety property in the abstraction is reproduced by the concrete continuous system.

Discrete abstractions may have transitions that are impossible (i.e. *spurious*) in the continuous system. In what follows we will describe the process of

constructing a sound abstraction of a given continuous system in a way that eliminates the impossible transitions between discrete states in the abstraction. That is, the resulting abstraction will feature a discrete transition between two abstract states if and only if a corresponding continuous trajectory is possible in the concrete system. We will use the underlying ideas to formulate an automatic method for synthesizing semi-algebraic continuous invariants.

5.2.1 Constructing the discrete state space

The first step in computing a discrete abstraction of a continuous system $\dot{\vec{x}} = f(\vec{x})$ & H concerns the construction of the discrete state space for the abstract discrete transition system. This process is known as *discretisation*, where the object being *discretised* is the evolution domain constraint H.

Discretisation is achieved by partitioning (in the standard mathematical sense) the evolution constraint H using *predicates*, which is the origin of the term *predicate abstraction* used by some authors [TK02] and traditionally applied to abstractions of systems that are already discrete (see e.g. [GS97]). The predicates we will consider are given by sign conditions on polynomial functions.

Definition 103. *A* **semi-algebraic decomposition** *of a set $H \subseteq \mathbb{R}^n$ by a finite set of polynomials $A \subset \mathbb{R}[x_1, x_2, \ldots, x_n]$ is a partition of H into k regions described by* non-empty *intersections of H with conjunctive formulas giving the possible combinations of sign conditions on all the polynomials in A.*

Computing the semi-algebraic decomposition of the evolution domain constraint H using a finite set of polynomials A can be achieved using a simple algorithm that we call *SemiAlgDecomp* (Algorithm 1).

Remark 104. Recall that we use the same notation for semi-algebraic sets and quantifier-free formulas of real arithmetic characterising them.

To apply the algorithm, we fix an ordering on the polynomials in A (thus turning A into a finite list) and compute *SemiAlgDecomp*$(\{H\}, A)$. The result is a finite set which defines a semi-algebraic decomposition of H by polynomials in A. We view this semi-algebraic decomposition as defining a partition of H into $k =$

Algorithm 1: *SemiAlgDecomp*

Data: $S_0 \subseteq 2^{\mathbb{R}^n}$, $A = (p_1, \ldots, p_m)$
Result: $S \subseteq 2^{\mathbb{R}^n}$

```
1  S ← {} ;
2  if |A| = 0 or S_0 = ∅ then
3      foreach s in S_0 do
4          if s ≠ ∅ then
5              S ← S ∪ {s} ;
6      return S
7  else
8      foreach s in S_0 do
9          S ← S ∪ {s ∧ (p_1 > 0)} ;
10         S ← S ∪ {s ∧ (p_1 = 0)} ;
11         S ← S ∪ {s ∧ (p_1 < 0)} ;
12     return SemiAlgDecomp(S, (p_2, ..., p_m))
```

$|\mathit{SemiAlgDecomp}(\{H\}, A)|$ regions, each corresponding to a single discrete state, which we denote by $\mathbf{s}_i$, where $1 \leq i \leq k$.

Algorithm 1 checks for non-emptiness of semi-algebraic sets (on line 4), which is a problem with exponential time complexity in the number of variables (see [BPR06, Chapter 3]) and can be solved using a decision procedure for the existential theory of the reals ($\exists\mathbb{R}$). In the worst case the algorithm will perform 3^m of these checks; in practice one can expect many of them to return False (i.e. when the conjunction of polynomial sign conditions has no solution over the reals). Note also that one may apply non-emptiness checks *in parallel* to all the states $\mathbf{s} \in S_0$.

5.2.2 Constructing the transition relation

Our goal in this section is to construct a transition relation on the states S (obtained from discretising a continuous system using Algorithm 1) in which a *single* transition from a discrete state $\mathbf{s}_i$ to another discrete state $\mathbf{s}_j$ is possible if (and only if) there is a trajectory in the continuous system starting inside the region corresponding to $\mathbf{s}_i$ and entering the region corresponding to $\mathbf{s}_j$ *without first visiting any other states*. The set of discrete states together with such a transition

relation constitutes a sound and exact discrete abstraction of the continuous system.

To give the reader some intuition, below we will illustrate the process of constructing the discrete abstraction using an example where a single polynomial p is used for the discretisation; we then proceed to describe the general setting in which the continuous system is discretised using a finite set of m polynomials.

First, suppose that we are given some system $\dot{\vec{x}} = f(\vec{x})$ & H and a single polynomial function $p : \mathbb{R}^n \to \mathbb{R}$. Let us also assume for simplicity that H is given by $\mathbb{R}^n$ (i.e. there is no evolution constraint). By performing a semi-algebraic decomposition of H by polynomials in the set $A = \{p\}$, we partition H into 3 basic semi-algebraic sets characterised by $p > 0$, $p = 0$ and $p < 0$ (let us further assume that none of these define empty sets), each corresponding to a discrete state. Formally, one may write this as $|\mathit{SemiAlgDecomp}(\{H\}, A)| = 3$ and the three discrete states $\mathbf{s}_i \in \mathit{SemiAlgDecomp}(\{H\}, A)$ are given by sign conditions on the polynomial p.

Observe that a continuous solution of the differential equation cannot pass from a discrete state where $p > 0$ (for some polynomial $p \in A$) to a state where $p < 0$ without passing through $p = 0$ first, nor vice versa. Using this intuition, we can give a general definition of what it means for two discrete states to be *neighbouring*.

Definition 105. *Let S be the set of discrete states constructed from a semi-algebraic set H and a list of polynomials $A = (p_1, \ldots, p_m)$. Two discrete states $\mathbf{s}_i, \mathbf{s}_j \in S$, where $i \neq j$, are neighbouring if their union (in the concrete state space) does* **not** *contain points $\vec{x}_1, \vec{x}_2 \in \mathbf{s}_i \cup \mathbf{s}_j$ such that $p(\vec{x}_1) < 0$ and $p(\vec{x}_2) > 0$ for any polynomial $p \in A$.*

We begin building the discrete abstraction by introducing a *neighbouring transition relation* $T_n \subseteq S \times S$ for the set of discrete states S in which every pair of neighbouring states has transitions in both directions between them. Intuitively, in the neighbouring transition relation one cannot "jump across" $p = 0$ in a single discrete transition; at the same time, any state is reachable from any other state. The initial abstraction which results from (S, T_n) is maximally coarse and

therefore not very useful.

We are only interested in those discrete transitions for which the corresponding continuous transitions are possible in the original continuous system. We eliminate impossible discrete transitions by *deciding* continuous invariance assertions. In this simple example we only have the following four:

$$p > 0 \rightarrow [\ \dot{\vec{x}} = f(\vec{x})\ \&\ H \wedge (p > 0 \vee p = 0)\ ]\ p > 0, \tag{5.1}$$

$$p < 0 \rightarrow [\ \dot{\vec{x}} = f(\vec{x})\ \&\ H \wedge (p < 0 \vee p = 0)\ ]\ p < 0, \tag{5.2}$$

$$p = 0 \rightarrow [\ \dot{\vec{x}} = f(\vec{x})\ \&\ H \wedge (p > 0 \vee p = 0)\ ]\ p = 0, \tag{5.3}$$

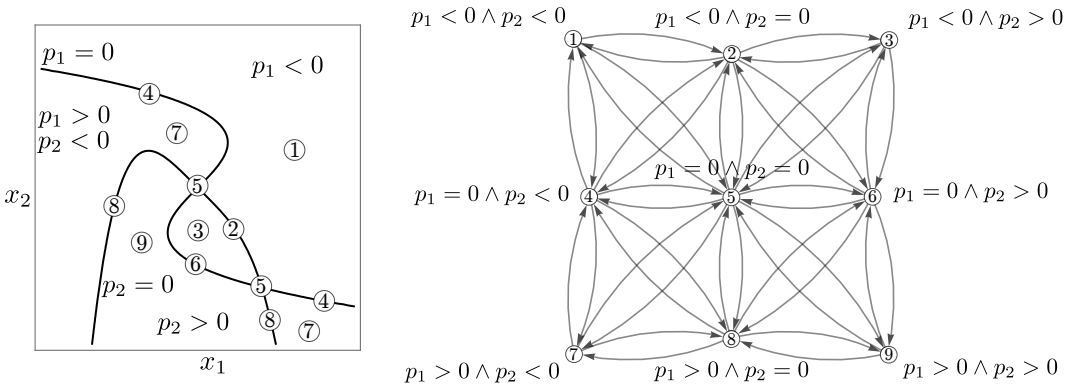
$$p = 0 \rightarrow [\ \dot{\vec{x}} = f(\vec{x})\ \&\ H \wedge (p < 0 \vee p = 0)\ ]\ p = 0. \tag{5.4}$$

We proceed by removing transitions from T_n as follows: if formula (5.1) is True, then we remove the transition from state $(p > 0)$ to state $(p = 0)$; otherwise, the transition $(p > 0) \longrightarrow (p = 0)$ is retained.

Remark 106. Note that by using a decision procedure for continuous invariant assertions [LZZ11], a continuous trajectory in the concrete system that starts in a state $\vec{x}_0 \in H$ where $p(\vec{x}) > 0$ and evolves into a state $\varphi_t(\vec{x}_0) \in H$ where $p(\varphi_t(\vec{x}_0)) = 0$ is possible *if and only if* the decision procedure returns False for the assertion given by formula (5.1).

Formula (5.2) is analogous but concerns transitions from $(p < 0)$ to $(p = 0)$. If formula (5.3) is False, we retain the transition $(p = 0) \longrightarrow (p > 0)$; otherwise the transition is removed. Similarly, formula (5.4) determines transitions from $(p = 0)$ to $(p > 0)$. When two polynomials p_1, p_2 are used for the abstraction, the number of discrete transitions one needs to validate is at most 9 (shown in Figure 5.1).

Now we turn to the general case where discretisation is performed using a finite set of m polynomials and completely determine the transition relation $T \subseteq S \times S$ for the set of discrete states S. We achieve this by considering the continuous flow between neighbouring regions corresponding to the discrete states without explicitly solving the ODEs, using the decision procedure for continuous invariant checking. By computing the transition relation T in this way, we arrive at a discrete abstraction of the continuous system, given by the pair (S, T), that is *exact* with respect to the discretisation. In what follows, we will write $\mathbf{s}_i \longrightarrow \mathbf{s}_j$



(a) Discretization S.

(b) Transition system (S, T_n).

Figure 5.1: Semi-algebraic decomposition of $\mathbb{R}^2$ by $A = \{p_1, p_2\}$ resulting in 9 discrete states $S \subset 2^{\mathbb{R}^2}$ and the neighbouring transition relation $T_n \subset S \times S$.

for $(\mathbf{s}_i, \mathbf{s}_j) \in T$, the transition between discrete states $\mathbf{s}_i, \mathbf{s}_j \in S$ in the transition relation T.

For each pair of neighbouring discrete states $(\mathbf{s}_i, \mathbf{s}_j) \in T_n$, by considering the invariance assertion

$$\mathbf{s}_i \rightarrow [\ \dot{\vec{x}} = f(\vec{x})\ \&\ (\mathbf{s}_i \vee \mathbf{s}_j)\ ]\ \mathbf{s}_i$$

we will proceed to remove transitions $\mathbf{s}_i \longrightarrow \mathbf{s}_j$ from the neighbouring transition relation T_n if and only if the decision procedure for continuous invariance assertions returns True.

Remark 107. The existence of a transition from a state $\mathbf{s}_i$ to a neighbouring state $\mathbf{s}_j$ is equivalent to the validity of the d$\mathcal{L}$ formula

$$\exists \vec{x}.\ \mathbf{s}_i \wedge \langle\ \dot{\vec{x}} = f(\vec{x})\ \&\ (\mathbf{s}_i \vee \mathbf{s}_j)\ \rangle\ \mathbf{s}_j,$$

which in d$\mathcal{L}$ is equivalent to the invalidity of the formula

$$\forall \vec{x}.\ \mathbf{s}_i \rightarrow [\ \dot{\vec{x}} = f(\vec{x})\ \&\ (\mathbf{s}_i \vee \mathbf{s}_j)\ ]\ \neg \mathbf{s}_j,$$

which, in turn, reduces to the invalidity of

$$\mathbf{s}_i \rightarrow [\ \dot{\vec{x}} = f(\vec{x})\ \&\ (\mathbf{s}_i \vee \mathbf{s}_j)\ ]\ \mathbf{s}_i.$$

One can view the process of removing impossible transitions as a sound refinement of the neighbouring transition relation T_n to the exact transition relation $T \subseteq T_n$.

This process can mechanised by a terminating procedure given in Algorithm 2, where (in the interest of saving space) we denote by *NeighbourTrans*(S) the procedure for constructing the neighbouring transition relation T_n for the discrete states S.

Algorithm 2: *ExactAbstraction*

Data: $\dot{\vec{x}} = f(\vec{x})$ & H, $A = (p_1, \ldots, p_m)$
Result: Discrete abstraction (S, T)

```
1 S ← SemiAlgDecomp({H}, A) ;
2 T_n ← NeighbourTrans(S) ;
3 T ← T_n ;
4 foreach (s_i, s_j) in T_n do
5 |   if s_i → [ẋ = f(x) & (s_i ∨ s_j)] s_i then
6 |   |   T ← T \ {(s_i, s_j)} ;
7 return (S, T)
```

Since T_n is finite and the formula in the conditional (on line 5) can be decided, Algorithm 2 is guaranteed to remove all impossible discrete transitions from the neighbouring transition relation and results in a new discrete transition system (S, T) that does not feature impossible transitions; we will state this property formally.

Proposition 108. *Abstractions (S, T) do not feature impossible transitions and are thus* **exact** *with respect to the discretisation. More formally, $\mathbf{s}_i \longrightarrow \mathbf{s}_j$ is in T if and only if*

$$\exists\, \vec{x}_0 \in \mathbf{s}_i.\ \exists\, \tau > 0.\ \varphi_0(\vec{x}_0) \in \mathbf{s}_i \ \wedge\ \varphi_\tau(\vec{x}_0) \in \mathbf{s}_j \text{ and}$$
$$\forall\, \tau \in [0, t].\ \varphi_t(\vec{x}_0) \in \mathbf{s}_i \cup \mathbf{s}_j,$$

that is, if and only if the system may evolve continuously from state $\mathbf{s}_i$ into a neighbouring state $\mathbf{s}_j$ without leaving their union $\mathbf{s}_i \cup \mathbf{s}_j$. The abstraction is thus exactly as coarse as the partition of the evolution domain constraint H into regions corresponding to the discrete states.

The most expensive step performed in Algorithm 2 is deciding the invariance assertion in the conditional on line 5. The implementation of computable conditions for continuous invariants [LZZ11] requires both checking membership

in ideals generated by multivariate polynomials (this problem is exponential space complete [May97] and can be solved using Gröbner bases) and deciding sentences in the theory of real arithmetic.

Remark 109. In the *worst-case*, using a list of m polynomials to perform the discrete abstraction will result in a neighbouring transition relation T_n with a total of $7^m - 3^m$ discrete transitions that need to be validated. To see this, consider a transition relation on 3^m discrete states (obtained from the discretisation by m polynomials) in which each state is connected to every state (including itself); in this case, we obtain $|3^m \times 3^m| = 9^m$ discrete transitions. Now, if we rule out all discrete transitions that cannot be neighbouring, i.e. of the form $(p_i > 0) \longrightarrow (p_i < 0)$ and $(p_i < 0) \longrightarrow (p_i > 0)$, where p_i is any polynomial used to construct the discrete state space, we obtain the following transition table with 7 transitions

	$p_i > 0$	$p_i = 0$	$p_i < 0$
$p_i > 0$	$\times$	$\times$	$-$
$p_i = 0$	$\times$	$\times$	$\times$
$p_i < 0$	$-$	$\times$	$\times$

Each discrete state can be viewed as an ordered list of polynomial sign conditions $(p_1 \sim 0, p_2 \sim 0, \ldots, p_m \sim 0)$, where $\sim \in \{<, =, >\}$. The transition table above can be used to iterate through this list to generate the transition relation for all the combinations of sign conditions that correspond to the discrete states. This relation will thus have $\underbrace{7 \times 7 \times \cdots \times 7}_{m} = 7^m$ transitions. If we further disregard all *stuttering* (also known as *self-looping* [SKHP96]) transitions, of which there are 3^m, we obtain $7^m - 3^m$ as our upper bound for the number of discrete transitions in T_n.

In practice, the number of discrete transitions in T_n will often be much lower. Furthermore, removing impossible transitions from T_n is a massively parallel problem and in practice one may exploit multi-core parallelism instead of iterating through the transitions sequentially, as in Algorithm 2.

Note also that it is possible to remove further impossible transitions from T_n when $m > 1$. We do not pursue this, since it would obscure the presentation and have no effect on the overall result; however, further performance improvements are possible, even though the worst-case bound on the number of neighbouring transitions will remain exponential in the number of polynomials.

5.2.3 Coarseness and unsoundness in existing approaches

Discrete abstraction of continuous systems by basic semi-algebraic predicates has previously been studied in [TK02, Tiw08a], where a simple abstraction method is proposed. We will now discuss some important differences between this earlier work and our approach.

The discrete abstraction method reported in [Tiw08a] is fundamentally different in the way it constructs the transition relation (let us call it $T_\sim \subseteq S \times S$), which is described in [Tiw08a, Section 3.2.2]. In essence, the method imposes conditions for removing transitions from the neighbouring transition relation T_n in the following way: given two neighbouring states $\mathbf{s}_i$ and $\mathbf{s}_j$, it removes the transition $\mathbf{s}_i \longrightarrow \mathbf{s}_j$ from T_n if *any* of the following conditions are satisfied for any $p \in A$:

- $\mathbf{s}_i$ contains $p < 0$ and $\mathbf{s}_j$ contains $p = 0$ and $\mathbf{s}_i \rightarrow \frac{dp}{dt} \leq 0$ is true,
- $\mathbf{s}_i$ contains $p > 0$ and $\mathbf{s}_j$ contains $p = 0$ and $\mathbf{s}_i \rightarrow \frac{dp}{dt} \geq 0$ is true,
- $\mathbf{s}_i$ contains $p = 0$ and $\mathbf{s}_j$ contains $p < 0$ and $\mathbf{s}_i \rightarrow \frac{dp}{dt} = 0$ or $\mathbf{s}_i \rightarrow \frac{dp}{dt} > 0$ is true,
- $\mathbf{s}_i$ contains $p = 0$ and $\mathbf{s}_j$ contains $p > 0$ and $\mathbf{s}_i \rightarrow \frac{dp}{dt} = 0$ or $\mathbf{s}_i \rightarrow \frac{dp}{dt} < 0$ is true.

Remark 110. The method described in [Tiw08a] also considers so-called *stuttering* transitions $\mathbf{s}_i \longrightarrow \mathbf{s}_i$, which we disregard (already in the way we define T_n). This discrepancy makes no practical difference to safety verification as stuttering transitions have no effect on the reachable sets of discrete abstractions where *dwell-time* in an abstract state is of no concern.

The approach in [Tiw08a] is not (in general) sound when the abstraction polynomials in A are allowed to be non-linear. To see this, consider the system

$$\dot{x}_1 = x_1, \quad \dot{x}_2 = x_2$$

evolving inside $H = \mathbb{R}^2$ and apply the method to perform the abstraction using a single polynomial $p = (x_1^2 + x_2^2 - 1)^3$. This results in a discrete transition system in which there is a transition from $(p < 0)$ to $(p = 0)$,

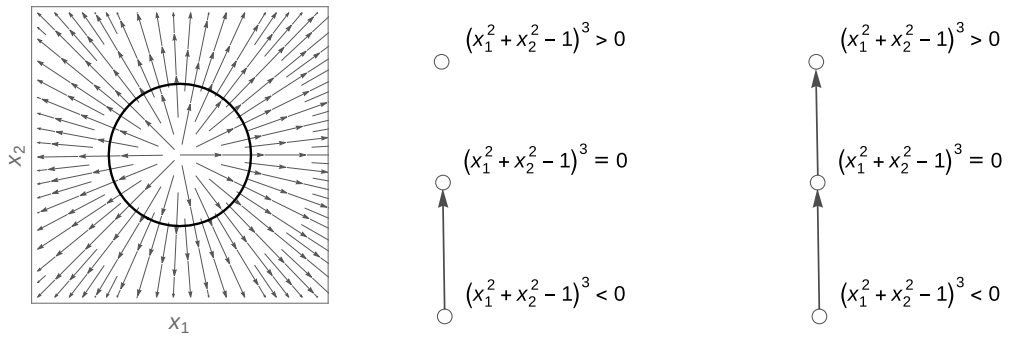
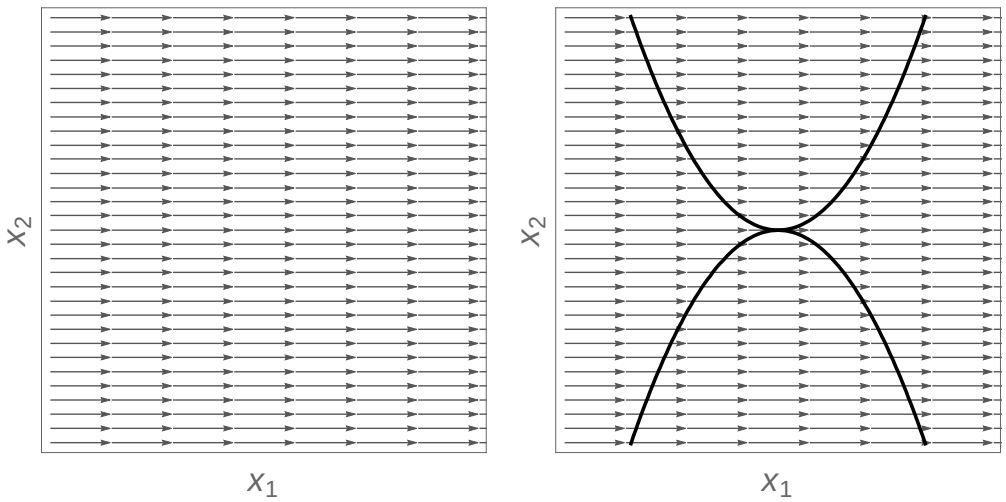


Figure 5.2: (**left**) Phase portrait and curve $(x_1^2 + x_2^2 - 1)^3 = 0$, shown in black. (**centre**) Unsound abstraction using method from [Tiw08a] and (**right**) sound abstraction using our approach.

but no transition from $(p = 0)$ to $(p > 0)$. One may thus conclude that starting inside $(p < 0)$ one cannot reach $(p > 0)$, which is not true for the continuous system. The source of unsoundness lies in the use of only the *first* derivatives to determine the transition behaviour in states where $p = 0$, while assuming the state constraint H. A simple way to fix this problem would be to impose a *smoothness condition* requiring that $\nabla p \neq \vec{0}$ when $p = 0$. However, this will remove unsoundness only in the special case when there is only *one* polynomial p used for the abstraction (and when the constraint H is an open set).



(a) Phase portrait for $\dot{x}_1 = 1, \dot{x}_2 = 0$. (b) Curves $x_1^2 + x_2 = 0$ and $x_2 - x_1^2 = 0$.

Figure 5.3: Counterexample for abstraction method in [Tiw08a].

In cases when there are two or more polynomials, even with the smoothness condition in place, the abstraction will not be sound in general. For example, consider the unconstrained system with constant derivatives $\dot{x}_1 = 1, \dot{x}_2 = 0$ and let $A = \{x_1^2 + x_2, x_2 - x_1^2\}$. The abstraction one obtains (Figure 5.4) suggests that the state $x_1^2 + x_2 = 0 \wedge x_2 - x_1^2 = 0$ (which is equivalent to $x_1 = 0 \wedge x_2 = 0$ over $\mathbb{R}$) is invariant under the flow of the system, which is incorrect.

The nature of the problem is more involved and also affects the approach to checking closed semi-algebraic invariants in [TT09], discussed in Chapter 4. Our sound and exact abstraction is shown in Figure 5.5.

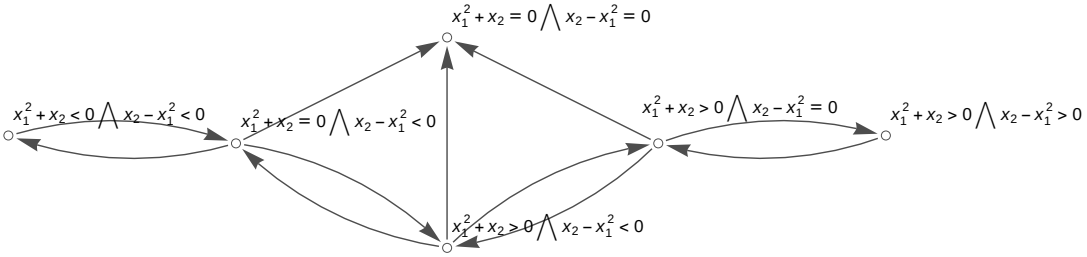


Figure 5.4: Abstraction using method from [Tiw08a] with extra smoothness condition.

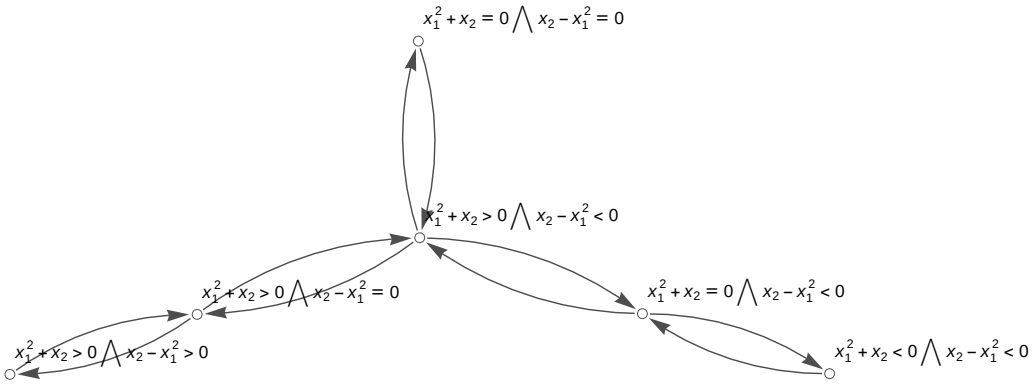


Figure 5.5: Exact abstraction (S, T) using our approach.

Apart from soundness issues, the method in [Tiw08a] suffers from coarseness, because it can introduce impossible discrete transitions. For instance, consider a planar system of non-linear ordinary differential equations

$$\dot{x}_1 = -x_1^3 - x_2^2 x_1 + x_1 + x_2,$$
$$\dot{x}_2 = -x_2^3 - x_1^2 x_2 + x_2 - x_1$$

featuring a stable limit cycle enclosing a point of equilibrium at the origin. Let the system evolve under no evolution constraints and consider a discretisation by the axes polynomials, i.e. take $A = \{x_1, x_2\}$, as illustrated in Figure 5.6.

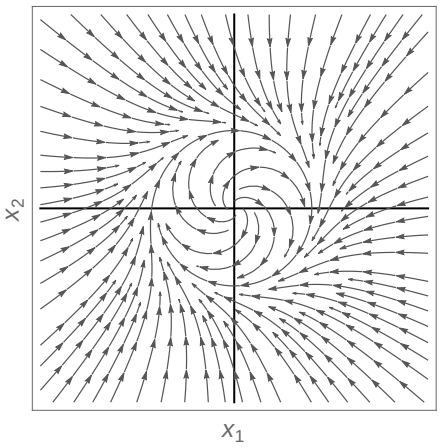
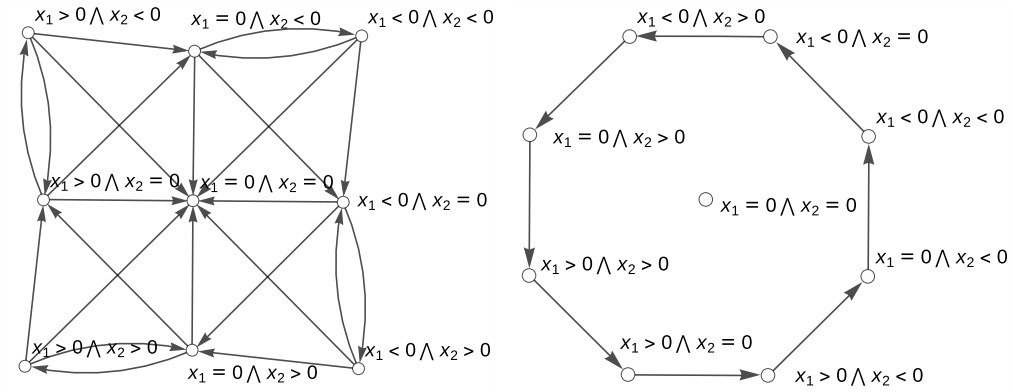


Figure 5.6: State space partition using $A = \{x_1, x_2\}$.

The discrete abstraction $(S, T_\sim)$ generated using the method from [Tiw08a] is shown in Figure 5.7, alongside an exact abstraction (S, T) generated using our approach. The abstraction $(S, T_\sim)$ suggests that the origin is reachable, while in the exact abstraction (S, T) the origin is correctly identified as an isolated invariant set.



(a) Abstraction method from [Tiw08a]. (b) Exact abstraction (S, T).

Figure 5.7: Removing coarseness.

5.3 Extracting continuous invariants from discrete abstractions

If one builds the full discrete abstraction (S, T) for a system $\dot{\vec{x}} = f(\vec{x})$ & H using some finite list of polynomials A, one may attempt to verify the safety of the system by showing that the safety property holds in the abstraction. For this, one needs to check whether an unsafe abstract state (i.e. one which contains a point that satisfies the formula $\neg\phi$) is reachable by following the discrete transitions in T starting from the initial set of states (those containing satisfiable instances of ψ). If none of the unsafe abstract states are reachable in the abstraction, one can conclude that the continuous system is safe.

By computing the forward-reachable set from the set of the initial states ψ in the abstraction, which we denote by $Reach_A^{\rightarrow}(\psi, H) \subseteq H$, one can generate a continuous invariant, which is the *smallest* continuous invariant with respect to the discrete abstraction by polynomials in A and is furthermore *semi-algebraic*. Formally, we define

$$Reach_A^{\rightarrow}(\psi, H) \equiv \bigvee_{\substack{i \text{ s.t. } \mathbf{s}_i \cap \psi \neq \emptyset, \\ j \text{ s.t. } \mathbf{s}_i \longrightarrow^* \mathbf{s}_j}} \mathbf{s}_j \ ,$$

where $\longrightarrow^*$ represents the reachability relation; that is, $\mathbf{s}_i \longrightarrow^* \mathbf{s}_j$ if state $\mathbf{s}_j$ is reachable from $\mathbf{s}_i$ in zero or more discrete transitions in the abstraction (T, S), obtained from the discretisation by polynomials in A. Thus, computing $I \equiv Reach_A^{\rightarrow}(\psi, H)$ yields a semi-algebraic set that is (by construction) guaranteed to include the initial set (i.e. $\psi \rightarrow I$) and is a continuous invariant for the system (i.e. $I \rightarrow [\dot{\vec{x}} = f(\vec{x})$ & $H]\ I$). If it is also true that I does not include any unsafe states (i.e. $I \rightarrow \phi$), then I is sufficient to conclude that the system is safe using the proof rule (*Safety*).

For invariant generation we are merely interested in extracting a semi-algebraic continuous invariant containing the initial set of states ψ from the abstraction and not the full abstraction (S, T) itself.

However, by computing the abstraction (S, T) in full, it is necessary to validate *every* transition in the neighbouring transition relation T_n and only then proceed to compute the reachable set $Reach_A^{\rightarrow}(\psi, H)$. Note that none of the transitions between the discrete states lying outside $Reach_A^{\rightarrow}(\psi, H)$ are ever used

in constructing the continuous invariant. The issue here lies in building the full abstraction *eagerly* and disregarding the set of initial states ψ. Indeed, this can afford a certain advantage, i.e. once the full abstraction is computed, one could supply an *arbitrary* semi-algebraic set of initial states ψ and quickly extract its reachable set from the abstraction. However, if the initial set of states ψ is already prescribed, then computing the full abstraction involves a lot of unnecessary computation with calls to expensive decision procedures.

Below we give a worklist Algorithm 3 for constructing the set $Reach_A^{\rightarrow}(\psi, H)$ that works by validating transitions between neighbouring discrete states *lazily* [1], i.e. on demand, while computing the set of reachable discrete states.

Algorithm 3: *LazyReach*

Data: $\psi, \dot{\vec{x}} = f(\vec{x})$ & H, A
Result: $Reach_A^{\rightarrow}(\psi, H)$

```
1  S ← SemiAlgDecomp({H}, A) ;
2  T_n ← NeighbourTrans(S) ;
3  Visited ← {s ∈ S | s ∩ ψ ≠ ∅} ;
4  Processed ← {} ;
5  while |Processed| < |Visited| do
6      Unprocessed ← Visited \ Processed ;
7      Processed ← Visited ;
8      foreach s_i in Unprocessed do
9          Validate ← {(s_i, s_j) ∈ T_n | s_j ∉ Visited};
10         foreach (s_i, s_j) in Validate do
11             if ¬(s_i → [ẋ = f(x) & (s_i ∨ s_j)] s_i) then
12                 Visited ← Visited ∪ {s_j} ;
13 return ⋁_{s ∈ Visited} s
```

This procedure, which we call *LazyReach*, proceeds as follows: firstly, as before, one computes the set of discrete states S and the neighbouring transition relation T_n (lines 1 and 2). The next step is to include the set of initial discrete states (i.e. those $\mathbf{s} \in S$ which intersect ψ) in the set of *visited* states (on line 3)[2]. Then, by making a distinction between discrete states that have merely been visited and

[1] One may regard Algorithm 3 as implementing a greedy search strategy in looking for an unsafe discrete state starting from a given set of initial states.

[2] Note that on lines 3 and 9 we employed the set builder notation to simplify presentation. In both cases, the operations required to construct these sets are sufficiently elementary.

those discrete states whose outgoing transitions have been *processed* (initially none, line 4), one can isolate the set of visited states that require processing.

This step is performed at each iteration (line 6) of the processing loop (lines 5-12). The main body of the loop then proceeds to gather all the discrete transitions from all the unprocessed visited states to all of their neighbouring states which are as yet unvisited (lines 8-9) and validates these transitions using a decision procedure for semi-algebraic continuous invariants (line 11). If a transition is not impossible, then the discrete state reachable using this transition is added to the set of visited states (on line 12) for processing in the next iteration.

The procedure saturates when no more unvisited states are reachable, and one obtains the reachable set $Reach_A^{\rightarrow}(\psi, H)$ by taking the union of all the visited states (line 13). In practice, employing Algorithm 3 results in fewer calls to the decision procedure for continuous invariance assertions than would be necessary if one were to compute the abstraction in full.[3]

Example 111 *Continuous invariant from reachable set*
Consider the system

$$\begin{aligned}\dot{x}_1 &= \left(x_1^2 - 1\right)\left(x_1^2 - \left(\sqrt{5} - 2\right)^2\right)\left(x_1 + \sqrt{5}x_2\right),\\ \dot{x}_2 &= \left(\sqrt{5}x_1 + x_2\right)\left(x_2^2 - 1\right)\left(x_2^2 - \left(\sqrt{5} - 2\right)^2\right),\end{aligned}$$

with $H = \mathbb{R}^2$ from [DLA06, Exercise 10.6, page 281]. As an initial set, suppose we take $\psi \equiv (x_1 - 1)^2 + x_2^2 < \frac{1}{4}$. Consider an abstraction of this system using the irreducible polynomial factors of the right-hand side of the system of ODEs, i.e. let

$$\begin{aligned}A = \{&x_1 - 1, x_1 + 1, x_1 - \sqrt{5} + 2, x_1 + \sqrt{5} - 2, -x_2 + \sqrt{5} - 2,\\ &x_2 - 1, x_2 + 1, x_2 + \sqrt{5} - 2, \sqrt{5}x_1 + x_2, x_1 + \sqrt{5}x_2\}.\end{aligned}$$

Algorithm 3 generates the following semi-algebraic continuous invariant:

$$\begin{aligned}&Reach_A^{\rightarrow}(\psi, H) = x_1 + 2 > \sqrt{5}\\ &\wedge \Big(\Big(x_2 + 1 > 0 \wedge x_2 + \sqrt{5} < 2 \wedge \Big(x_1 + \sqrt{5}x_2 \geq 0 \vee \sqrt{5}x_1 + x_2 > 0\Big)\Big)\\ &\quad \vee \Big(x_2 < 1 \wedge \Big(\Big(x_1 + \sqrt{5}x_2 > 0 \wedge x_2 + \sqrt{5} \geq 2\Big) \vee x_2 + 2 \geq \sqrt{5}\Big)\Big)\Big).\end{aligned}$$

[3]However, owing to its iterative nature, *LazyReach* may not exploit the advantage offered by parallel architectures to the full when validating discrete transitions.

■

Although the transitions in *Validate* (line 12) can be processed (lines 13-15) in parallel at every iteration of the main processing loop, the size of *Validate* may be much smaller than the number of cores available in the system. On the other hand, using a hypothetical system with $|T_n|$ cores, one could validate each transition in T_n in parallel.

The above approach to invariant generation avoids introducing new variables into the problem and furthermore only requires one to decide *universally quantified* sentences in the theory of real arithmetic, which is a less expensive problem than that of performing real quantifier elimination (see e.g. [BPR06, Chapter 13]).

5.4 Tackling state space explosion

Discrete abstractions of continuous systems suffer from the discrete state explosion problem, i.e. the number of discrete states in the abstraction grows exponentially with the number of polynomials $m = |A|$ used for the discretisation. This section will explore approaches to tackling the discrete state space explosion in order to improve scalability *without affecting the granularity* of the abstraction. Reducing the size of A (the list of polynomials used for the abstraction) is an obvious avenue for optimisation. However one would ideally wish to achieve this without making the abstraction any coarser. For this we will use *differential cuts* and *differential divide-and-conquer*.

It is clear that keeping a low number of discrete states in the abstraction is crucial if one wishes to work with more than the very basic examples. By reducing the number of discrete states one also dispenses with many discrete transitions and the computational cost incurred from validating them. As was remarked earlier, the number of discrete states in the abstraction grows (in the worst case) exponentially with the number of polynomials $m = |A|$ used to partition the evolution constraint. This clearly presents a scalability challenge and raises questions about the practicality of discrete abstraction.

5.4.1 Using differential cuts

In Chapter 4 we have already encountered a special form of differential cuts and showed how it can be used to make certain invariant checking problems involving equations more tractable. In their most general form, as introduced by Platzer and Clarke in [PC08], differential cuts were used for iteratively refining the evolution constraint H with differential invariants (a subset of continuous invariants, see Chapter 4). In its full generality, the proof rule DC is given by

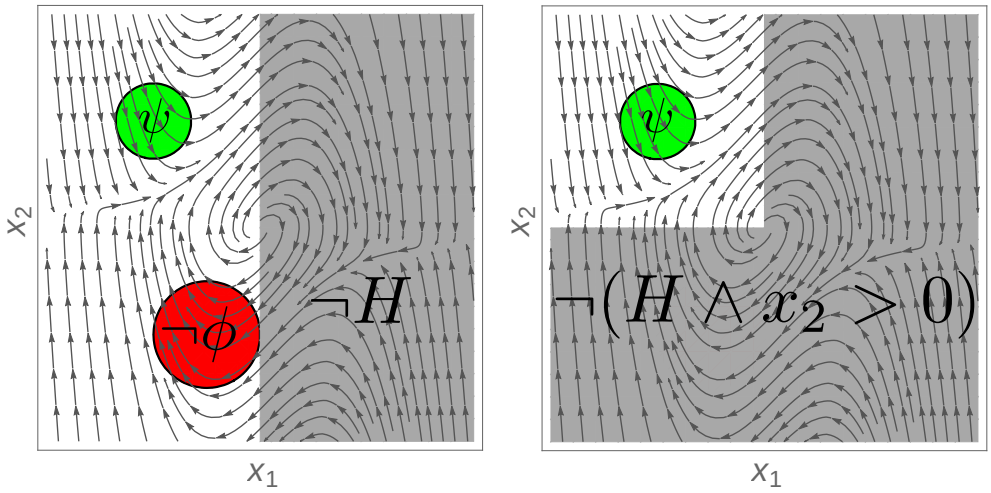
$$(\text{DC})\frac{\psi \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]F \qquad \psi \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H \wedge F]\,\phi}{\psi \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\;\phi}.$$

The rule formalises the idea that it is always sound to restrict the evolution domain H by some continuous invariant F, provided that it includes the initial set ψ, the original rationale being that it may be possible to prove the safety property using differential invariants in the more restricted system in the right premise.

Example 112 *Differential cut*
Consider the Van der Pol system (with $\mu = 1$)

$$\begin{aligned}\dot{x}_1 &= x_2,\\ \dot{x}_2 &= \left(1 - x_1^2\right) x_2 - x_1,\end{aligned}$$

under the evolution constraint $H \equiv x_1 < 0$. Suppose we are given a safety verification problem where the precondition ψ defines some region inside H where $x_2 > 0$ is true and the set of unsafe $\neg\phi$ states lies inside a region of H where $x_2 \leq 0$ holds, as illustrated in Figure 5.8a on page 123.



(a) Safety verification problem (b) Refined constraint.

Figure 5.8: Refining the evolution constraint H with DC.

In this particular system, the set characterized by $x_2 > 0$ is a continuous invariant that contains all the initial states ψ; therefore we may apply the rule DC with $F \equiv x_2 > 0$ to refine the evolution constraint to $H \wedge x_2 > 0$, obtaining a safety verification problem for a system (shown in Figure 5.8b on page 123) whose reachable set is confined within a *smaller* constraint. ■

In discrete abstraction, if one is to consider each individual polynomial $p \in A$, it is intuitive that if one can show that (**i**) for the initial set of states ψ, the polynomial p is sign-invariant, i.e. $p(\psi) \sim 0$ where $\sim \in \{<, =, >\}$ and (**ii**) that this sign condition defines a continuous invariant for the system, i.e. $p \sim 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ p \sim 0$, then one can *refine* the evolution constraint to $H \wedge p \sim 0$ and *remove* the polynomial p from A and obtain an abstraction by the polynomials $B \equiv A \setminus \{p\}$ which has the property that

$$Reach_B^{\rightarrow}(\psi, H \wedge p \sim 0) \equiv Reach_A^{\rightarrow}(\psi, H).$$

The number of discrete states one has when using B for discretizing $H \wedge p \sim 0$ is at most 3^{m-1} and the process can be repeated for other polynomials that remain in B.

5.4.2 Differential divide-and-conquer

We now introduce a proof rule akin to DC, that exploits a property of sets that are continuous invariants in both positive and negative time directions to split

the continuous system into *two* continuous sub-systems between which there is no continuous evolution. The principle at work here is essentially the same as that of the following basic theorem from the theory of dynamical systems.

Theorem 113. *Given a continuous system defined on some differentiable manifold M, a set S is positively invariant if and only if its complement $M \setminus S$ is negatively invariant under the flow. The set S is invariant (in both positive and negative time) if and only if $M \setminus S$ is invariant.*

Proof. Elementary. See [BS70, Chapter II, Theorem 1.4]. □

Proposition 114. *The proof rule* DDC *(differential divide-and-conquer) is sound.*

$$F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\, F \qquad\qquad \neg F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\, \neg F$$

$$\text{(DDC)}\frac{\psi \wedge F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H \wedge F]\ \phi \qquad \psi \wedge \neg F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H \wedge \neg F]\ \phi}{\psi \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\ \phi}$$

Proof. As a first step, we show that the following proof rule is sound:

$$\frac{\psi \wedge F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\ \phi \qquad \psi \wedge \neg F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\ \phi}{\psi \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\ \phi}$$

This follows from the fact that any formula F can be used to partition the pre-condition ψ into two disjoint regions $\psi \wedge F$ and $\psi \wedge \neg F$ whose union is exactly the pre-condition, i.e. $(\psi \wedge F) \vee (\psi \wedge \neg F) \equiv \psi$. Thus, given any initial state $\vec{x}_0 \models \psi$, we are guaranteed that either $\vec{x}_0 \models \psi \wedge F$ or $\vec{x}_0 \models \psi \wedge \neg F$. In both cases, the system is required to be safe by the premise and the soundness of the rule follows.

The second step is to apply differential cut (DC) to the two respective sub-goals:

$$\text{(DC)}\ \frac{\psi \wedge F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\ F \qquad \psi \wedge F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H \wedge F]\ \phi}{\psi \wedge F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\ \phi}$$

$$\text{(DC)}\ \frac{\psi \wedge \neg F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\ \neg F \qquad \psi \wedge \neg F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H \wedge \neg F]\ \phi}{\psi \wedge \neg F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\ \phi}$$

We now apply the following sound rules to the first sub-goal in the resulting proof branches, respectively

$$\frac{\psi \wedge F \to F \quad F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\; F}{\psi \wedge F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\; F} \qquad \frac{\psi \wedge \neg F \to \neg F \quad \neg F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\; \neg F}{\psi \wedge \neg F \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\; \neg F}$$

and use the fact that $\psi \wedge F \to F$ and $\psi \wedge \neg F \to \neg F$ are valid formulas to arrive at open sub-goals that are precisely those appearing in the premise of DDC, from which we conclude that DDC is sound. □

Informally, the rule allows one to split the original system into two disjoint regions that are not connected by a continuous flow, since both F and its complement $\neg F$ are continuous invariants. In the parlance of dynamical systems one would simply say that the rule splits the system into two sub-systems that evolve inside disjoint *invariant sets*.

Note that unlike DC, the rule DDC does *not* require the initial set ψ to be wholly contained inside F. Instead, DDC splits the initial set of states into two disjoint initial subsets $\psi \wedge F$ and $\psi \wedge \neg F$. The rule DDC thus decomposes the original safety assertion into two independent safety assertions about *smaller sub-systems*, which can be proved separately.

The following proposition (due to Ghorbal) goes a step further and works to split the safety verification problem into *three* smaller independent safety verification sub-problems. This is achieved by partitioning the evolution domain constraint using invariant algebraic varieties, i.e. invariants of the form $p = 0$, which can be checked using a variety of methods discussed in Chapter 4 (also see [GSP15a]).

Proposition 115 *(Ghorbal).* *The proof rule* $\text{DDC}_=$ *is sound.*

$$(\text{DDC}_=)\frac{\begin{array}{c} p = 0 \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\, p = 0 \quad p = 0 \to [\dot{\vec{x}} = -f(\vec{x}) \;\&\; H]\, p = 0 \\ \bigwedge_{\sim \in \{<,=,>\}} \Big(\psi \wedge p \sim 0 \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H \wedge p \sim 0]\; \phi\Big) \end{array}}{\psi \to [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\; \phi}$$

Proof. The evolution constraint H can be partitioned into 3 disjoint regions $H \wedge p > 0$, $H \wedge p = 0$ and $H \wedge p < 0$. If $p = 0$ defines an invariant set for the system (a consequence of the first two premises of $\text{DDC}_=$), then a trajectory

can neither enter nor leave $H \wedge p = 0$. Because p is a continuous function, any trajectory inside H initialised in a state where $p > 0$ holds and entering a state where $p < 0$ is true necessarily crosses the real variety defined by $p = 0$; an impossibility. Similarly, no trajectory may cross from $p < 0$ to $p > 0$, from which we conclude that the three disjoint regions are not connected by the continuous flow (i.e. their reachable sets under the flow are entirely disjoint) and may be treated as evolution constraints of independent continuous systems. The union of the reachable sets of the resulting sub-systems gives the reachable set of the original system. Therefore, if each sub-system is safe, one may conclude that the original system is safe. □

The rule $\text{DDC}_{=}$ is interesting from a practical perspective because it may be used in concert with automatic procedures for generating invariant *polynomial equalities* for polynomial ODEs (e.g. using methods developed by Ghorbal and Platzer in [GP14a]) to effectively reduce the original safety verification problem into smaller sub-problems.

We now turn to applying the rules DC and $\text{DDC}_{=}$ to tackle the state space explosion problem. In Algorithm 4 we give a procedure for refining the evolution domain constraint and removing polynomials from A whenever this is possible. We call this procedure *DWC* as it also exploits the reasoning principle of *differential weakening* DW [Pla10a]

$$\text{(DW)}\ \frac{H \to \phi}{\psi \to [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ \phi},$$

which simply requires that the evolution domain be contained within the post-condition to conclude that the system is safe, together with the proof rules DC and DDC.

On lines 3 and 4, *DWC* applies the rule DW as a sufficiency check for termination. On lines 7, 9 and 11 the procedure discards those p for which $p > 0$, $p < 0$ or $p = 0$ describe a continuous invariant containing the initial set ψ (conditionals on lines 6, 8 and 10). This step corresponds to an application of the rule DC with $F \equiv p > 0$, $F \equiv p < 0$ and $F \equiv p = 0$ which, if the rule application is successful, are used to refine the evolution constraint H in the recursive call. If $p = 0$ is an invariant in both positive and negative time and does *not* contain all the initial states ψ, one can use the proof rule DDC to work with 3 smaller sub-systems

Algorithm 4: *DWC*

Data: $\psi, \dot{\vec{x}} = f(\vec{x})\ \&\ H, \phi, A$

Result: Continuous invariant I s.t. $\psi \subseteq I$

1 **if** $H \wedge \psi \rightarrow$ False **then**
2 **return** False
3 **if** $H \rightarrow \phi$ **then**
4 **return** H //DW
5 **foreach** $p \in A$ **do**
6 **if** $(H \wedge \psi \rightarrow p > 0) \wedge \Big(p > 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ p > 0\Big)$ **then**
7 **return** $DWC(\psi, \dot{\vec{x}} = f(\vec{x})\ \&\ H \wedge p > 0, \phi, A \setminus \{p\})$ //DC
8 **if** $(H \wedge \psi \rightarrow p < 0) \wedge \Big(p < 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ p < 0\Big)$ **then**
9 **return** $DWC(\psi, \dot{\vec{x}} = f(\vec{x})\ \&\ H \wedge p < 0, \phi, A \setminus \{p\})$ //DC
10 **if** $(H \wedge \psi \rightarrow p = 0) \wedge \Big(p = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ p = 0\Big)$ **then**
11 **return** $DWC(\psi, \dot{\vec{x}} = f(\vec{x})\ \&\ H \wedge p = 0, \phi, A \setminus \{p\})$ //DC
12 **foreach** $p \in A$ **do**
13 **if** $(p = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ p = 0) \wedge (p = 0 \rightarrow [\dot{\vec{x}} = -f(\vec{x})\ \&\ H]\ p = 0)$ **then**
14 $GT \leftarrow DWC(\psi \wedge p > 0, \dot{\vec{x}} = f(\vec{x})\ \&\ H \wedge p > 0, \phi, A \setminus \{p\})$;
15 $EQ \leftarrow DWC(\psi \wedge p = 0, \dot{\vec{x}} = f(\vec{x})\ \&\ H \wedge p = 0, \phi, A \setminus \{p\})$;
16 $LT \leftarrow DWC(\psi \wedge p < 0, \dot{\vec{x}} = f(\vec{x})\ \&\ H \wedge p < 0, \phi, A \setminus \{p\})$;
17 **return** $GT \vee EQ \vee LT$ //DDC
18 **return** H

of the original system whose reachable set may be constructed by *combining the reachable sets of these smaller systems*. This idea is implemented on lines 13-17 of Algorithm 4, where *DWC* recurses on the 3 smaller sub-systems and removes the polynomial p (used to divide the system) from A. The over-approximations of reachable sets obtained using these 3 recursive calls are then combined into a union (line 17), which gives an over-approximation of the reachable set for the original system. Finally, when no further progress can be made, the procedure returns the evolution constraint H (line 18). Because the procedure only involves applying sound proof rules, one may view *DWC* as a *proof strategy* that can be implemented in a theorem prover. Indeed, if the procedure returns a result while there are still polynomials remaining in A, one has a proof of safety involving only the proof rules DW, DC and DDC.

Unlike *LazyReach*, the invariant generation procedure *DWC* will not (in general) always be able to find a sufficiently strong continuous invariant to prove the safety property, even if one exists in the semi-algebraic abstraction by the polynomials in A. The invariants *DWC* is able to generate are thus generally coarser than those generated using *LazyReach*. However, we observe that in the worst case the running-time of *DWC* is only quadratic in the number of polynomials $m = |A|$, i.e. $\mathcal{T}_{DWC}(m) = O(m^2)$, compared the exponential time complexity of *LazyReach*.

Consider now a combination of the procedure *DWC* together with the *LazyReach* algorithm. Algorithm 5 gives an invariant generation procedure that we call *DWCL*, which is obtained simply by replacing the final line (18) in *DWC* with

$$\textbf{return } \mathit{LazyReach}(\psi, \dot{\vec{x}} = f(\vec{x}) \;\&\; H, A).$$

Algorithm 5: *DWCL*

Data: $\psi, \dot{\vec{x}} = f(\vec{x}) \;\&\; H, \phi, A$
Result: Continuous invariant I s.t. $\psi \subseteq I$

1 **if** $H \wedge \psi \rightarrow$ False **then**
2 **return** False
3 **if** $H \rightarrow \phi$ **then**
4 **return** H //DW
5 **foreach** $p \in A$ **do**
6 **if** $(H \wedge \psi \rightarrow p > 0) \wedge \left(p > 0 \rightarrow [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\; p > 0\right)$ **then**
7 **return** $DWC(\psi, \dot{\vec{x}} = f(\vec{x}) \;\&\; H \wedge p > 0, \phi, A \setminus \{p\})$ //DC
8 **if** $(H \wedge \psi \rightarrow p < 0) \wedge \left(p < 0 \rightarrow [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\; p < 0\right)$ **then**
9 **return** $DWC(\psi, \dot{\vec{x}} = f(\vec{x}) \;\&\; H \wedge p < 0, \phi, A \setminus \{p\})$ //DC
10 **if** $(H \wedge \psi \rightarrow p = 0) \wedge \left(p = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\; p = 0\right)$ **then**
11 **return** $DWC(\psi, \dot{\vec{x}} = f(\vec{x}) \;\&\; H \wedge p = 0, \phi, A \setminus \{p\})$ //DC
12 **foreach** $p \in A$ **do**
13 **if** $(p = 0 \rightarrow [\dot{\vec{x}} = f(\vec{x}) \;\&\; H]\; p = 0) \wedge (p = 0 \rightarrow [\dot{\vec{x}} = -f(\vec{x}) \;\&\; H]\; p = 0)$ **then**
14 $GT \leftarrow DWC(\psi \wedge p > 0, \dot{\vec{x}} = f(\vec{x}) \;\&\; H \wedge p > 0, \phi, A \setminus \{p\})$;
15 $EQ \leftarrow DWC(\psi \wedge p = 0, \dot{\vec{x}} = f(\vec{x}) \;\&\; H \wedge p = 0, \phi, A \setminus \{p\})$;
16 $LT \leftarrow DWC(\psi \wedge p < 0, \dot{\vec{x}} = f(\vec{x}) \;\&\; H \wedge p < 0, \phi, A \setminus \{p\})$;
17 **return** $GT \vee EQ \vee LT$ //DDC
18 **return** $\mathit{LazyReach}(\psi, \dot{\vec{x}} = f(\vec{x}) \;\&\; H, A)$ //*LazyReach*

Instead of returning H when no further progress can be made with *DWC*, *DWCL* falls back to using the more expensive *LazyReach* algorithm with the remaining

polynomials. This combined procedure is theoretically as powerful as *LazyReach*, i.e. is capable of extracting the exact reachable set $Reach_A^{\rightarrow}(\psi, H)$ if necessary, but in practice also as fast as *DWC*, although theoretically the running time of *DWCL* remains exponential in m.

Example 116 *Continuous invariant generated using DWCL*
Consider the system

$$\dot{x}_1 = 2x_1\left(x_1^2 - 3\right)\left(4x_1^2 - 3\right)\left(x_1^2 + 21x_2^2 - 12\right),$$
$$\dot{x}_2 = x_2\left(35x_1^6 + 105x_2^2x_1^4 - 315x_1^4 - 63x_2^4x_1^2 + 378x_1^2 + 27x_2^6 - 189x_2^4 + 378x_2^2 - 216\right)$$

with $H = \mathbb{R}^2$ from [DLA06, Exercise 10.7, page 281]. As an initial set, suppose we take $\psi \equiv (x_1 - 1)^2 + x_2^2 < \frac{1}{4}$ and let $\phi \equiv x_1^2 + x_2^2 < 8$ be the post-condition. Consider an abstraction of this system using the irreducible polynomial factors of the right-hand side of the system of ODEs and the post-condition, i.e. let

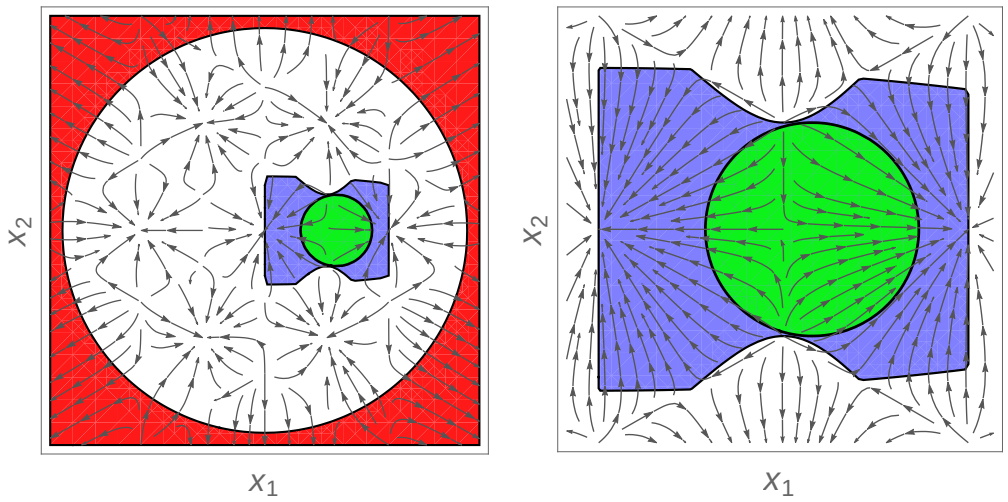
$$A = \{x_1, x_1^2 - 3, 4x_1^2 - 3, x_2, x_1^2 + x_2^2 - 8, x_1^2 + 21x_2^2 - 12,$$
$$35x_1^6 + 105x_2^2x_1^4 - 315x_1^4 - 63x_2^4x_1^2 + 378x_1^2 + 27x_2^6 - 189x_2^4 + 378x_2^2 - 216\}.$$

There are 7 abstraction polynomials in total, which in the worst-case would lead to $3^7 = 2187$ discrete states and $7^7 - 3^7 = 821356$ discrete transitions in the neighbouring transition relation T_n. In practice, applying *LazyReach* to generate the reachable set $Reach_A^{\rightarrow}(\psi, H)$ for this problem takes an unreasonable amount of time. The procedure *DWC* takes significantly less time to run, but is unable to find a suitable invariant using DW, DC and $\text{DDC}_=$ alone. Our implementation of the combined procedure *DWCL* is able to generate the following continuous invariant $I \subset \phi$ (see Figure 5.9 on page 130) in 104 seconds: [4]

$$\left(4x_1^2 = 3 \wedge x_1 > 0 \wedge \left(x_2 = 0 \vee 35x_1^6 + 105\left(x_2^2 - 3\right)x_1^4 + 27\left(x_2^6 - 7x_2^4 + 14x_2^2 - 8\right) < 63x_1^2\left(x_2^4 - 6\right)\right)\right)$$
$$\vee \left(x_2 = 0 \wedge \left(0 < x_1 < \frac{\sqrt{3}}{2} \vee \frac{\sqrt{3}}{2} < x_1 < \sqrt{3}\right)\right)$$
$$\vee \Bigg(x_1^2 + 21x_2^2 < 12 \wedge 35x_1^6 + 105\left(x_2^2 - 3\right)x_1^4 + 27\left(x_2^6 - 7x_2^4 + 14x_2^2 - 8\right) < 63x_1^2\left(x_2^4 - 6\right)$$
$$\wedge \left(0 < x_1 < \frac{\sqrt{3}}{2} \vee \left(2x_1 > \sqrt{3} \wedge x_1^2 < 3 \wedge x_2 \neq 0\right)\right)\Bigg).$$

For this problem, the procedure *DWCL* makes repeated use of both DC and $\text{DDC}_=$ (each is used 4 times in total) before falling back to *LazyReach*, which

[4] expression simplified in *Mathematica*.



(a) Phase portrait, unsafe states $\neg\phi$ (**red**), initial set ψ (**green**).

(b) Enlarged view of the invariant and the initial set.

Figure 5.9: Automatically generated continuous invariant $I \subset \phi$ (**blue**).

in every instance is given 3 polynomials that remain to perform the abstraction (down from 7 in the original list A). ■

5.5 Selecting discretisation polynomials

Discrete predicate abstraction of continuous systems relies on the user supplying a set of polynomials A for discretising the continuous evolution constraint. In general, it is very difficult to assess the "quality" of an abstraction by only considering the discretisation polynomials in A; however, certain polynomials can easily be seen to be poor candidates. For instance, a polynomial function p that is sign-invariant inside the domain constraint H, i.e. such that $p(\vec{x})$ does not change sign for any $\vec{x} \in H$, clearly cannot be used to partition H further and is thus of no value.

In this section we discuss certain sources of polynomials for discrete abstraction. We will develop some intuition about the respective merits and limitations of the predicates that we can generate from these sources, giving brief illustrations.

5.5.1 Polynomials extracted from the problem

The verification problem itself is often a good source of polynomials; e.g. they could come from the description of the post-condition ϕ, the pre-condition ψ, or indeed from the right-hand side of the system of ODEs, i.e. the polynomials $f_1, f_2, \ldots, f_n$, their irreducible factors, etc [5].

While this approach may appear very simplistic, it is able to capture some important properties. For instance, by using the right-hand side of the ODEs, i.e. by setting $A = (f_1, f_2, \ldots, f_n)$, the abstraction is guaranteed to capture all equilibria in the system, which will appear as a discrete state with no outgoing transitions. Intuitively, this is obvious since $f_1 = 0 \wedge f_2 = 0 \wedge \cdots \wedge f_n = 0$ corresponds to a single discrete state in the abstraction at which there is no evolution in the concrete system.

In practice, we also observe that it is often a good choice to *factorise* the polynomials that one extracts from the problem and use their irreducible factors in the list A for the abstraction. Let us note that

$$p = p_1 p_2 \cdots p_r = 0 \quad \equiv_{\mathbb{R}} \quad p_1 = 0 \vee p_2 = 0 \vee \cdots \vee p_r = 0.$$

Using the irreducible factors p_i, $1 \leq i \leq r$, may intuitively be viewed as a refinement of the abstraction by the polynomial p because it serves to increase the number of discrete states and partitions the curve $p = 0$, allowing the abstraction to determine the transitions through each curve separately.

Example 117
Consider the planar non-linear system from [DLA06, Exercise 1.11]

$$\dot{x}_1 = -x_1^5 - x_2^4 x_1 + 2x_1,$$
$$\dot{x}_2 = -x_2^3 - x_1^2 x_2 + x_2.$$

The abstraction which results from using the right-hand side captures the point of equilibrium at the origin, but is otherwise not very useful (Figure 5.11). If one instead factorises the right-hand side, one obtains

$$A = \{-x_1, x_1^4 + x_2^4 - 2, -x_2, x_1^2 + x_2^2 - 1\},$$

[5] For linear and affine systems one could also extract polynomials corresponding to the hyperplanes generated from the eigenvectors of the dynamics, as in [HR08].

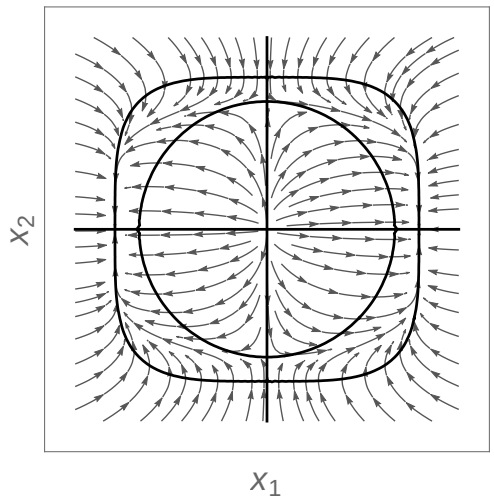


Figure 5.10: State space partition by $A = \{-x_1^5 - x_2^4 x_1 + 2x_1, -x_2^3 - x_1^2 x_2 + x_2\}$.

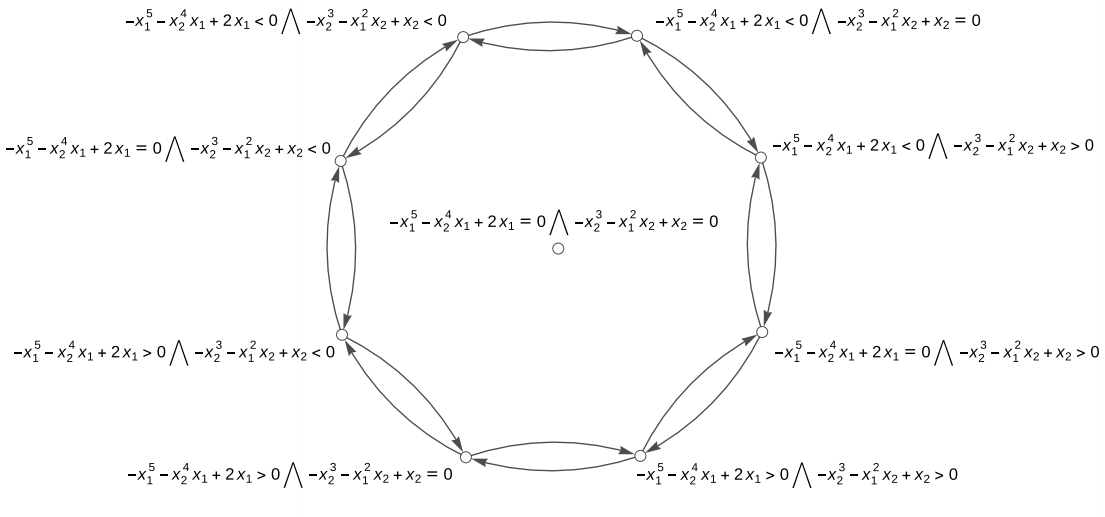


Figure 5.11: Exact abstraction using $A = \{-x_1^5 - x_2^4 x_1 + 2x_1, -x_2^3 - x_1^2 x_2 + x_2\}$.

which results in a larger, but much better abstraction (shown in Figure 5.12 on page 133). ■

5.5.2 Higher-order Lie derivatives

If one has collected a certain list of discretisation polynomials A, but the resulting abstraction is too coarse to prove the desired property, one typically needs to either generate a fresh list of polynomials for a new abstraction or add more polynomials into the existing list A. This will often result in a sizable increase in the number of discrete states, so one needs to be judicious in the choice of

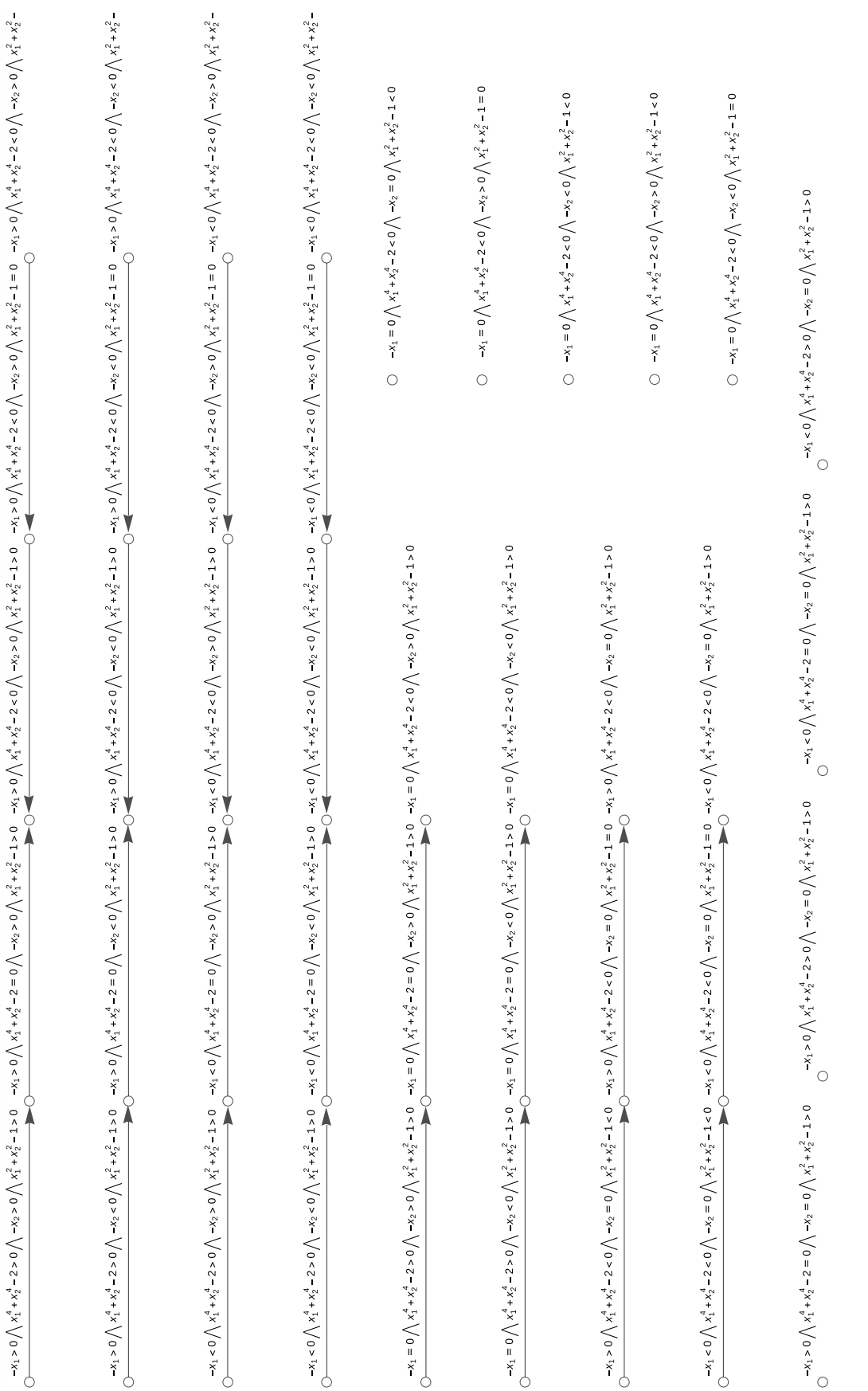


Figure 5.12: Abstraction using irreducible factors $A = \{-x_1, x_1^4 + x_2^4 - 2, -x_2, x_1^2 + x_2^2 - 1\}$.

polynomials with which to grow the list A. Lie derivatives of the polynomials already in A are common candidates for this task.

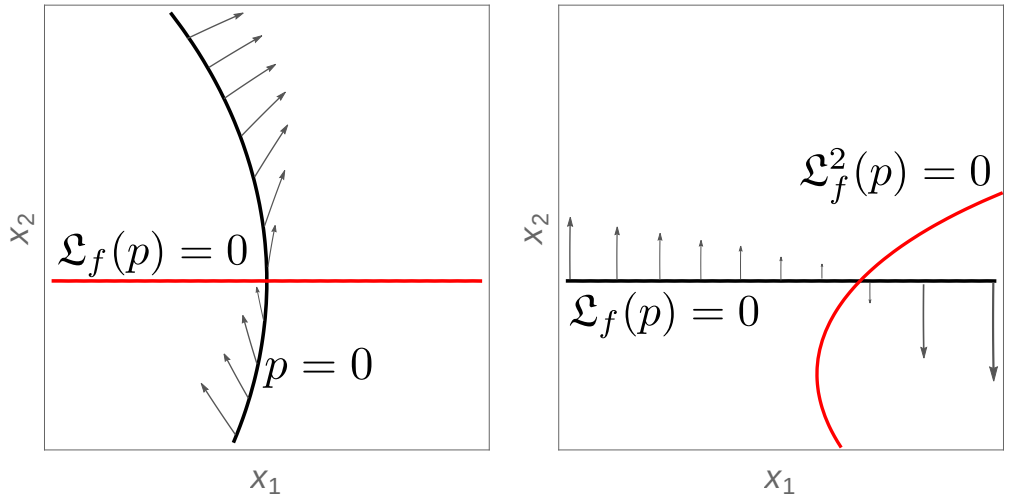


Figure 5.13: Geometric intuition behind Lie derivatives as abstraction polynomials.

One may appreciate the intuition behind adding the Lie derivative $\mathfrak{L}_f(p)$ of some polynomial $p \in A$ into the list by observing that the curves $p = 0$ and $\mathfrak{L}_f(p) = 0$ intersect at points where the flow of the continuous system across the boundary $p = 0$ can change direction (as illustrated in Figure 5.13). This process can be applied iteratively to the Lie derivatives themselves by adding $\mathfrak{L}_f^2(p) \equiv \mathfrak{L}_f(\mathfrak{L}_f(p))$, and so on, into A.

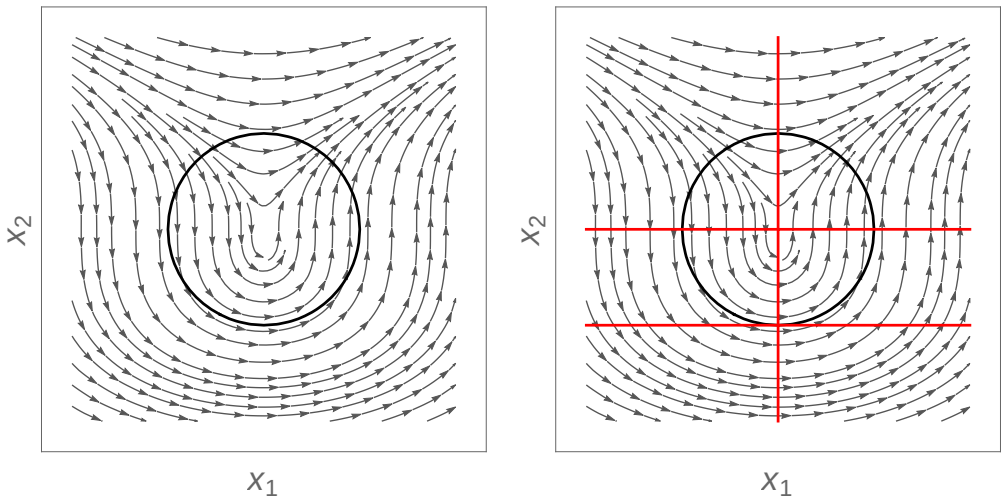
Remark 118. Note that in general the curves $p = 0$ and $\mathfrak{L}_f(p) = 0$ need not intersect.

Example 119

Consider the simple non-linear system

$$\dot{x}_1 = x_2^2,$$
$$\dot{x}_2 = x_1$$

and let $p = x_1^2 + x_2^2 - 1$. The variety of p is shown in black in Figure 5.14. Now consider the Lie derivative $\mathfrak{L}_f(p) = 2x_1x_2^2 + 2x_1x_2$; the variety of this polynomial (shown in red in Figure 5.14) intersects the unit circle at points where the continuous flow changes from "entering" to "exiting" the unit disc. Note that $\mathfrak{L}_f(p)$ factorises into $(2x_1)(x_2)(1 + x_2)$; adding these three factors separately into A results in a finer abstraction in which every discrete state featuring $p = 0$ is



(a) Phase portrait and $p = 0$ (in black). (b) $p = 0$ and $\mathfrak{L}_f(p) = 0$ (in red).

Figure 5.14: Lie derivative in the abstraction.

guaranteed to have the property that the direction of the continuous flow through the section of the curve $p = 0$ inside that state is invariant. ■

The use of higher-order Lie derivatives as a source for polynomials for discrete abstraction was previously investigated by Tiwari in [Tiw08a] (see also [TK04] for related work), where certain closure properties are used to formulate a heuristic for selecting "good" candidates to be included in the list A.

5.5.3 Darboux invariants and invariant real varieties

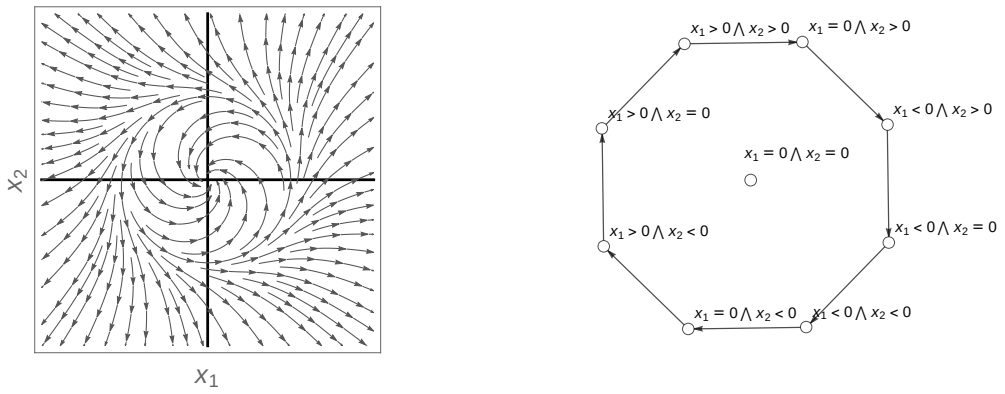
Zaki et al. investigated the use of Darboux polynomials (see Section 4.2.4) for performing discrete abstraction of polynomial ODEs in [ZTB06, ZTB07], where it was observed that Darboux invariants can often be used to identify *separatrices* in the system and are thus natural candidates for qualitative abstraction. Furthermore, abstraction using Darboux invariants is sound and does not induce coarseness because each state in the abstraction is guaranteed to be invariant (since each $p \in A$ is Darboux, there can be no incoming or outgoing discrete transitions in between states with $p > 0$, $p = 0$ or $p < 0$). We note that Darboux invariants provide only a sufficient condition for ensuring this property and one can extend this idea to a more general class of polynomials – those whose real zero sets define invariant real varieties (a property that can now also be checked [GP14a, LZZ11]).

Example 120

Consider again the system of non-linear ordinary differential equations:

$$\dot{x}_1 = x_1^3 + x_2^2 x_1 - x_1 - x_2,$$
$$\dot{x}_2 = x_2^3 + x_1^2 x_2 - x_2 + x_1.$$

Computing the discrete abstraction by $A_1 = \{x_1, x_2\}$ results in the discrete transition system shown below.



(a) State space partition by A. (b) Exact abstraction (S, T).

Figure 5.15: Simple abstraction using $A = \{x_1, x_2\}$.

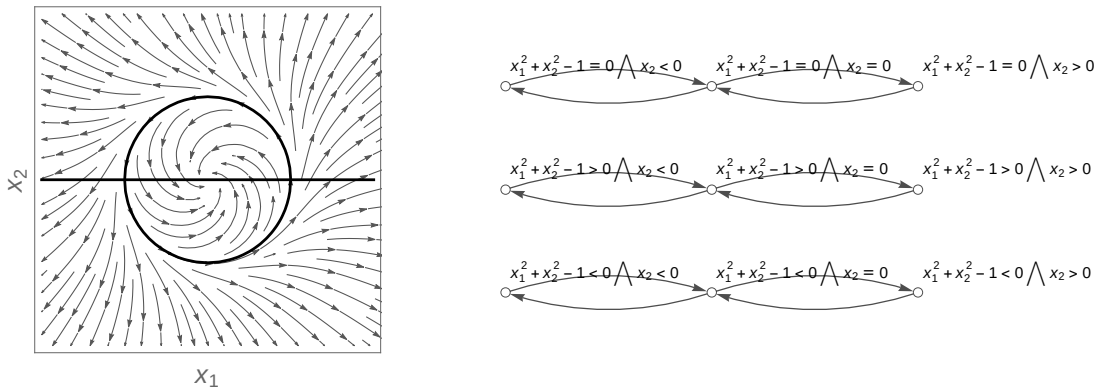
Now consider the same system with a discrete abstraction obtained from a different set of polynomials $A_2 = \{x_1^2 + x_2^2 - 1, x_2\}$. The variety corresponding to the unit circle is an invariant set for the system, since

$$\begin{aligned}\mathfrak{L}_f(x_1^2 + x_2^2 - 1) &= 2x_1\dot{x}_1 + 2x_2\dot{x}_2 \\ &= 2(x_1^2 + x_2^2)(x_1^2 + x_2^2 - 1) \\ &\in \langle x_1^2 + x_2^2 - 1 \rangle,\end{aligned}$$

i.e. because $x_1^2 + x_2^2 - 1$ is a Darboux polynomial for the system (see Theorem 58 in Section 4.2.4). Such a partition of the state space (shown in Figure 5.16) yields a different discrete transition system in which we can clearly observe three invariant sets, which may be treated as three separate sub-systems.

■

The description of an invariant variety in a system may not be apparent from the description of the system itself. For instance, in the example above, while clearly visible in the phase portrait, the description of the invariant unit circle was not



(a) State space partition by A. (b) Exact abstraction (S, T).

Figure 5.16: Abstraction using an invariant variety, $A = \{x_2, x_1^2 + x_2^2 - 1\}$.

immediately available from the polynomials in the right-hand side of the system of ODEs [6]. In practice, the description of invariant varieties in a system may be very far from obvious and it is highly desirable to have automatic means of searching for them.

5.6 Practical evaluation

In this section we compare the performance of our continuous invariant generation algorithms *LazyReach*, *DWC* and *DWCL* on a set of 100 safety verification problems for continuous systems (see Appendix A). The running time performance [7] of the algorithms is summarised in Figures 5.17, 5.18 and 5.19. For each experiment the problems have been sorted according to the running time for a decision or to a 600 second time-out and plotted with the problem number on the horizontal axis and the (log scale) running time on the vertical axis. By *solved* we understand that a semi-algebraic continuous invariant has been successfully generated and that it implies the postcondition, i.e. is sufficient to prove the safety assertion. In our experiments we:

1. use polynomial factors of the right-hand side of the ODEs together with the factors of the polynomials appearing in the postcondition ϕ to create the list of polynomials A (Figure 5.17),

[6] Unless one rewrites them as $\dot{x}_1 = -x_2 + x_1(x_1^2 + x_2^2 - 1)$, $\dot{x}_2 = -x_1 + x_2(x_1^2 + x_2^2 - 1)$.
[7] The comparison was performed on an *i5-3570K* CPU clocked at 3.40GHz.

2. extend the list A generated as in 1) with Lie derivatives of every polynomial in A (Figure 5.18), and

3. explore the utility of using polynomials whose real roots are invariant real algebraic sets by extending the list of polynomials generated in 1) with algebraic invariants generated using a method presented in [GP14a] (Figure 5.19).

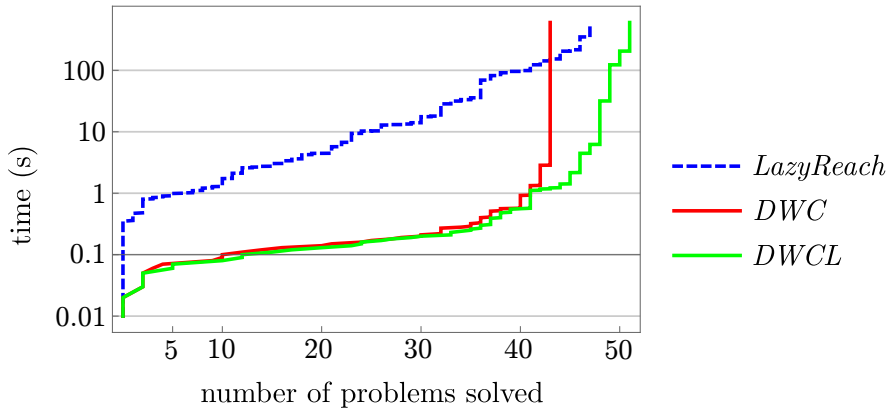


Figure 5.17: Safety verification performance using factors of polynomials in the ODEs and the post-condition to populate A.

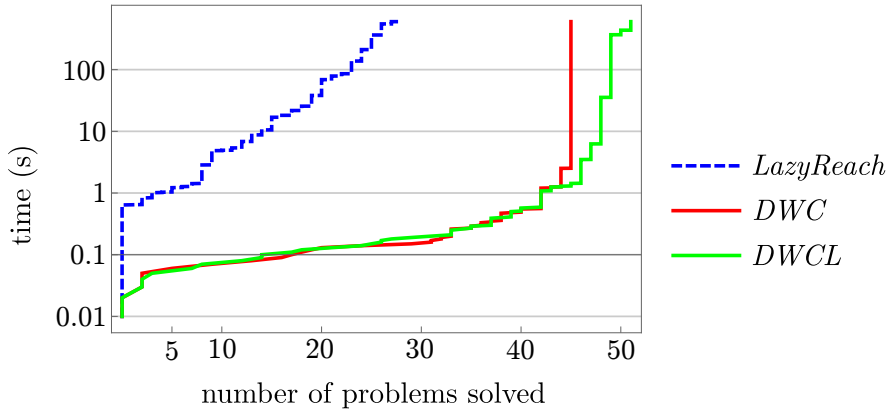


Figure 5.18: Safety verification performance using a larger set A, consisting of factors of polynomials in the ODEs, the post-condition ϕ and their (first order) Lie derivatives.

In our results we observe that the *DWC* algorithm is significantly faster than *LazyReach*, confirming our hopes for gains in efficiency. We also observe that, when using polynomial factors of the ODEs and the postcondition to abstract the system, *LazyReach* was able to prove more problems (47) than *DWC* (43). This is not surprising, since DW, DC and DDC, while very efficient, cannot in

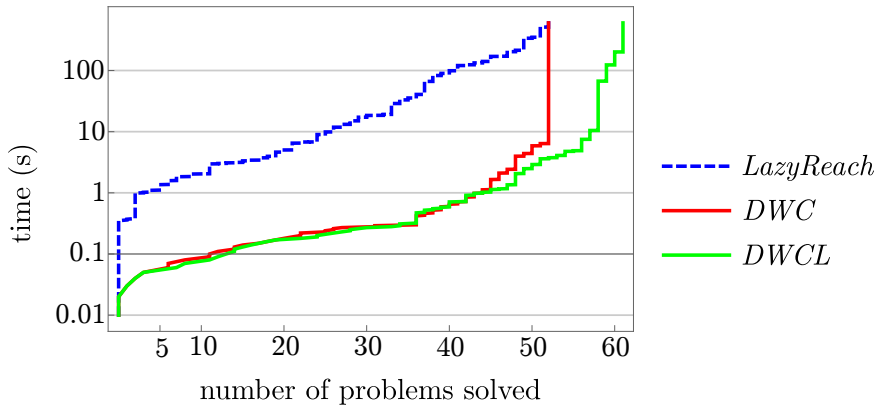


Figure 5.19: Safety verification performance using factors of polynomials in the ODEs, the post-condition and algebraic invariants to populate A.

general be used to extract reachable sets of exact abstractions like *LazyReach*. The combined method *DWCL* (using DW, DC, and DDC before falling back to *LazyReach*) is seen to be both as practically efficient as *DWC* and able to solve more problems (51) than *LazyReach* under a 600 second timeout; of course, given enough time, *DWCL* and *LazyReach* will both succeed at solving exactly the same problems (with *LazyReach* taking significantly more time).

Adding the first Lie derivatives of the polynomial factors of the ODE and the postcondition effectively doubles the size of the list A which, unsurprisingly, leads to diminished performance of *LazyReach* (only 28 problems solved) because it is heavily affected by the discrete state explosion problem. However, *DWC* is seen to perform slightly better than it did without the Lie derivatives in the list, solving a total of 45 safety verification problems. The *DWCL* algorithm succeeds at proving safety in 51 of the problems yet again, but the set of systems that are proved safe is different from that of *DWCL* when used with a list of polynomial factors of the ODE and the postcondition without the Lie derivatives.

We observe that adding algebraic invariants to the list of polynomial factors of the ODE and the postcondition resulted in a palpable improvement in the number of problems that could be solved. This is very clearly visible in the case of *DWC*, which is guaranteed to process every algebraic invariant by applying the proof rule DDC. Overall, for this choice of polynomials we see *LazyReach* and *DWC* both solving 52, and *DWCL* solving 61 problems out of 100.

These results are very encouraging as they demonstrate that the discrete state explosion problem can, to a certain extent, be addressed using algorithms such as *DWCL* and that methods for automatic algebraic invariant generation (such as that in [GP14a]) can be used to generate polynomials that will often improve the quality of the resulting abstractions.

5.7 Combined model checking-like approach

In this section we will briefly discuss a combination of the verification approach based on semi-algebraic abstraction, using methods developed earlier in this chapter, and methods for analysing bounded-time reachability using enclosures of bounded-time reachable sets of ODEs.

Discrete abstraction of ODEs can be used to solve safety verification problems by considering an over-approximation of the continuous reachable set for *unbounded time*. The main issue with discrete abstraction is the discrete state explosion problem. Though in principle one can use arbitrarily many polynomials for the abstraction, in practice the number of polynomials often needs to be modest. A small set of polynomials leads to a (semi-algebraic) reachable set in the abstraction that gives only a very coarse over-approximation of the reachable set in the continuous system.

An alternative model checking-like approach is to consider *bounded-time* safety verification by constructing *enclosures* that give a bounded-time over-approximation of the reachable set. Numerous enclosure construction methods have been proposed by various authors (an excellent survey may be found in [NJN07]). Very broadly, these fall into two classes: interval-based methods for *verified integration* of ODEs [NJC99] (tools include e.g. *VNODE-LP* by Nedialkov) and more recent approaches based on Taylor models [BM98] (implemented in e.g. *Flow** [CÁS12] by Chen et al. and *VSPODE* by Lin and Stadtherr [LS07]).

The chief limitation of bounded-time verification approaches is their inability (save for some special cases) to prove system safety for all future time. Furthermore, the initial set of states from which the enclosure is to be constructed

is required to be compact and connected [CSÁ14] (in practice often given by a closed hyperbox). Furthermore, the over-approximating enclosure can in practice be very precise for small time horizons, but tends to become conservative when the time bound is large (due to the so-called *wrapping effect*, which is a problem caused by the successive build-up of over-approximation errors that arises in interval-based enclosure construction approaches; see e.g. [NJN07]). Notwithstanding these theoretical limitations, tools based on enclosure construction have been applied to analyse bounded-time reachability of complicated non-linear systems in a number of impressive case studies (see e.g. [CÁS12]).

In order to combine discrete abstraction with bounded-time verification methods, we will require some auxiliary definitions.

Definition 121. *Given an n-dimensional system of ODEs $\dot{\vec{x}} = f(\vec{x})$ and an initial set of states $\psi \subseteq \mathbb{R}^n$, an* ***enclosure*** *of duration t, denoted $E(\psi, t)$, is an over-approximation of the reachable set (from ψ) for time t, i.e. $\{\varphi_\tau(\vec{x}) \mid \vec{x} \in \psi, \tau \in [0, t]\} \subseteq E(\psi, t)$.*

Remark 122. The definition above assumes that solutions $\varphi_\tau(\cdot)$ exist and are of sufficient duration (i.e. defined for $\tau \in (a, b)$, with $a < t < b$ and $a < 0$).

The concept of *weakest pre-condition* was originally introduced by Dijkstra [Dij75] for proving partial correctness properties of discrete programs using Hoare logic. Below we will (very loosely) adapt this concept to the analysis of continuous systems using exact discrete abstractions.

Definition 123. *Given a continuous system $\dot{\vec{x}} = f(\vec{x})$ & H, a pre-condition ψ, a post-condition ϕ and a set of polynomials A, we define the* **weakest pre-condition** *WP_A of the system in the semi-algebraic abstraction by polynomials in A as follows:*

$$WP_A(\dot{\vec{x}} = f(\vec{x}) \ \&\ H, \phi) \equiv H \setminus Reach_A^{\leftarrow}(\neg\phi, H),$$

where $Reach_A^{\leftarrow}(\cdot)$ is the **backward-reachable** *set in the abstraction, defined as $Reach_A^{\rightarrow}(\cdot)$ for the system in which the dynamics is reversed, i.e. $\dot{\vec{x}} = -f(\vec{x})$ & H.*

Proposition 124. *The weakest pre-condition* $WP_A(\dot{\vec{x}} = f(\vec{x}) \ \& \ H, \phi)$ *is a continuous invariant for the continuous system* $\dot{\vec{x}} = f(\vec{x}) \ \& \ H$.

Proof. There are no outgoing discrete transitions from the set $WP_A(\dot{\vec{x}} = f(\vec{x}) \ \& \ H, \phi)$ in the abstraction by A. A transition between two discrete states is present if and only if a continuous evolution between them is possible. Therefore, there can be no continuous flow out of the weakest pre-condition. □

Remark 125. Note that with our definition of the weakest pre-condition WP_A, one may soundly prove the safety assertion $\psi \to [\dot{\vec{x}} = f(\vec{x}) \ \& \ H] \ \phi$ by showing that $\psi \to WP_A(\dot{\vec{x}} = f(\vec{x}) \ \& \ H, \phi)$.

Proposition 126. *Starting from some initial set of states* ψ*, if after some finite time* $t_2 > 0$ *the enclosure entirely enters the weakest pre-condition, i.e. for some* $0 \leq t_1 < t_2$ *we have* $E(\psi, t_2) \setminus E(\psi, t_1) \subseteq WP_A(\dot{\vec{x}} = f(\vec{x}) \ \& \ H, \phi)$*, without reaching an unsafe state, i.e.* $E(\psi, t_2) \subseteq \phi$*, then (since the weakest pre-condition itself contains no unsafe states, i.e.* $WP_A(\dot{\vec{x}} = f(\vec{x}) \ \& \ H, \phi) \subseteq \phi$*) the system is safe.*

Proof. $E(\psi, t_2)$ is an over-approximation of the reachable set for time $t_2 > 0$ and for all $\vec{x} \in \psi$ we have either $\varphi_{t_2}(\vec{x}) \in WP_A(\dot{\vec{x}} = f(\vec{x}) \ \& \ H, \phi)$ or $E(\psi, t_1) = E(\psi, t_2)$. In the latter case, the reachable set did not advance and therefore $E(\psi, t_1)$ contains a continuous invariant that includes ψ and lies inside the post-condition and the system is therefore safe. In the former case, we have $E(\psi, t_2) \subseteq \phi$ and the weakest pre-condition is a continuous invariant for the system and contains no states satisfying $\neg\phi$, so we are guaranteed that $\varphi_t(\varphi_\tau(\vec{x})) \in \phi$ holds for all $\tau \in [0, t_2]$ and $t \geq 0$, therefore $\varphi_{\tau+t}(\vec{x}) \in \phi$ is true for all $(\tau + t) \geq 0$ and we conclude that the system is safe. □

We will illustrate the intuition behind this approach using a basic example.

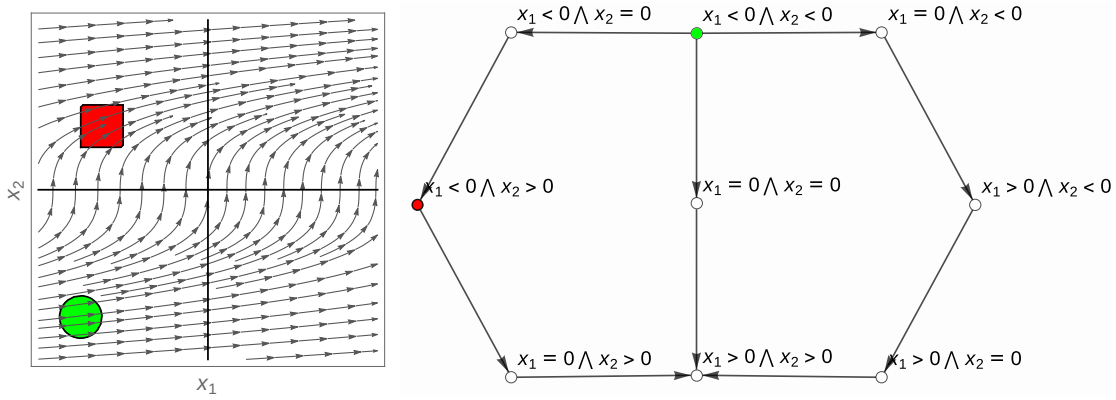
Example 127

Consider the following non-linear system evolving under no evolution constraints:

$$\dot{x}_1 = x_2^2, \quad \dot{x}_2 = 1, \quad H \equiv \mathbb{R}^2.$$

Let the set of initial states be given by $\psi \equiv (x_1 + 3)^2 + (x_2 + 3)^2 \leq \frac{1}{4}$ and the set of *unsafe* states be $\neg\phi \equiv x_2 > 1 \wedge x_2 < 2 \wedge x_1 < -2 \wedge x_1 > -3$. Suppose

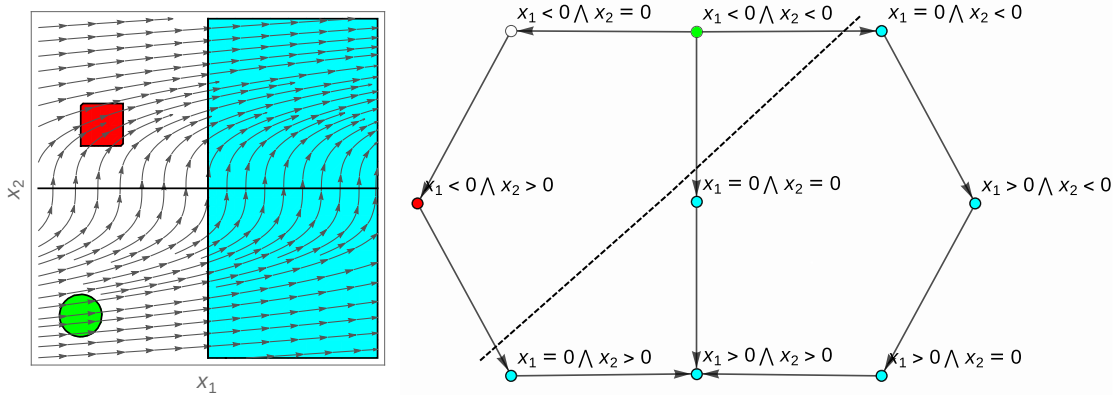
further that we select the axis polynomials to perform the discrete abstraction, i.e. we choose $A = \{x_1, x_2\}$. In Figure 5.20 we see that the abstraction alone is insufficient to prove safety.



(a) Safety verification problem. (b) Abstraction by polynomials $A = \{x_1, x_2\}$.

Figure 5.20: Safety verification in the abstraction. Initial states ψ shown in green, unsafe states $\neg\phi$ in red.

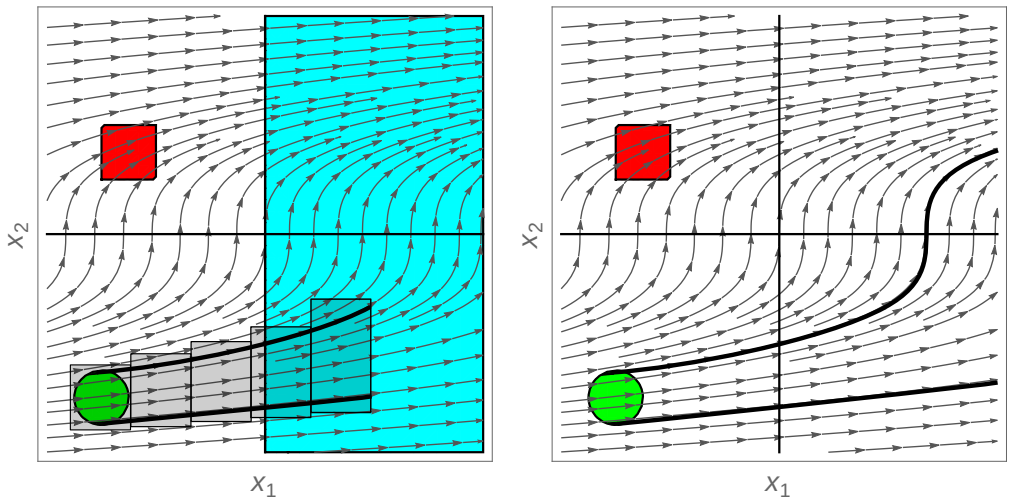
By computing the weakest pre-condition in the abstraction, it is obvious that it does not include the set of initial states (Figure 5.21). However, one may prove that the system is safe by computing an enclosure of the reachable set from ψ and showing that it enters the weakest pre-condition without reaching any unsafe states, i.e. applying Proposition 126 (illustrated in Figure 5.22).



(a) Safety verification problem. (b) Abstraction by polynomials $A = \{x_1, x_2\}$.

Figure 5.21: Safety verification in the abstraction. Initial states ψ shown in green, unsafe states $\neg\phi$ in red and the weakest pre-condition of ϕ in the abstraction by polynomials $A = \{x_1, x_2\}$, $WP_A(\dot{\vec{x}} = f(\vec{x}) \;\&\; H, \phi) \equiv x_1 \geq 0$ in cyan.

■



(a) Enclosure. (b) Forward reachable set.

Figure 5.22: Enclosure (light grey) reaching the weakest precondition of ϕ in the abstraction by $A = \{x_1, x_2\}$ (cyan) sufficient to prove safety.

Limitations The combined safety verification method described above inherits all the limitations that pervade existing bounded-time verification methods. For instance, one can no longer work with arbitrary semi-algebraic initial sets ψ, which are now required to be compact and connected. More fundamentally, the method does *not* produce an explicit semi-algebraic continuous invariant that can be used in a deductive prover to certify system safety independently within a sound proof calculus. Instead, one has to trust the implementation of the enclosure construction algorithm to be correct and the semi-algebraic abstraction to be sound in order to conclude system safety.

Despite these deficiencies, we believe that such a combined approach can be implemented into a powerful standalone tool for checking unbounded time safety assertions about ODEs. Such a tool would extend the capability of existing bounded-time safety verification software by incorporating qualitative reasoning based on sound semi-algebraic abstractions that we developed in this chapter. Furthermore, recent work on formal verification of enclosure construction algorithms by Immler [Imm14, Imm15] is a promising development that has the potential to dispel most concerns users might have about the soundness of using bounded-time enclosures of reachable sets of ODEs.

5.8 Related work

In [ZZK13], the authors apply their earlier results about checking semi-algebraic continuous invariants to address the invariant generation problem using approaches such as pre-defining parametric templates and restricting attention to classes of invariants (such as polyhedra), as well as using qualitative analysis techniques to suggest invariant templates. Our approach is different in that we do not rely on parametric templates and put no restrictions on the form of the semi-algebraic invariant which may be generated.

Discrete abstraction of *linear* continuous (and hybrid) systems using *linear polynomials* was studied in [ADI03, ADI06]. Abstractions of non-linear systems using more general polynomials were investigated in [TK02, Tiw08a]. We have presented a *sound* method for computing abstractions of non-linear systems by semi-algebraic predicates that is different from [TK02, Tiw08a] in determining the discrete transitions using a *decision procedure* for continuous invariant assertions [LZZ11]; we thus eliminate impossible transitions that are introduced by the method described in [Tiw08a]. This is a significant improvement because it entirely removes a major source of coarseness, as well as unsoundness.

Abstraction of continuous systems by *timed automata* was studied in [SKHP96, SEK99, SW11, SW13, MB08, CNL12], where additional restrictions are placed on the type of discretisation (e.g. by only employing sub-level sets of Lyapunov or Lyapunov-like functions [SW11, SW13], which are already difficult to obtain, or an orthogonal grid [SKHP96, SEK99, MB08, CNL12] to partition the continuous state space). The focus of abstractions by timed automata is often placed on demonstrating liveness properties, where the notion of time spent in an abstract state is crucial.

Combining elements of qualitative and quantitative reasoning to study the behaviour of dynamical systems has previously been explored in the case of planar systems by Nishida et al. [NMKD91]; however, we should note that the qualitative theory of dynamical systems is an incredibly broad subject that goes well beyond the study of safety properties of ODEs. The focus of our interest has been very specific; for a broader overview of previous work on the general qualitative analysis of dynamical systems, the interested reader is invited

to consult [Kui86, Sac90a, Sac90b, Zha94, NMKD91, LK93, Kok95, ABW97]. More recently, Carter [Car13] investigated so-called *deadness properties* of hybrid systems that can be used to disprove the temporal property of liveness. We may view the qualitative-quantitative safety verification approach described in this chapter as demonstrating deadness for the set of unsafe states. Similar ideas have also been explored by Clarke et al. in [CFH$^+$03], where polyhedral over-approximations of the flow are used to refine the abstraction.

5.9 Summary

Deductive verification tools for hybrid systems benefit greatly from the ability to prove continuous invariance assertions (as discussed in the previous chapter) and having the means of *automatically generating* continuous invariants sufficient to prove safety assertions about continuous systems. In practice, this latter point is often the main bottleneck when verifying safety of hybrid systems in which the continuous dynamics are non-linear. We have presented powerful techniques for automatically discovering continuous invariants and removed some important theoretical limitations (unsoundness and coarseness) in existing methods for constructing discrete abstractions of continuous systems.

Our interest in semi-algebraic discrete abstractions was motivated by the nature of their reachable sets, which define semi-algebraic continuous invariants for the original continuous system and can possess interesting boolean structure that cannot realistically be expected with invariants generated using existing template-based methods.

We view this work as bridging the two most prevalent approaches to safety verification of continuous systems: model checking-like approach (based on discrete abstraction) and the formal deductive verification approach (based on sound proof rules for checking continuous invariants). We have shown how a decision procedure for semi-algebraic continuous invariants (a central tool in the deductive approach) can be used to create finer abstractions than was previously possible and use these discrete abstractions (objects central to model checking continuous systems) to generate semi-algebraic continuous invariants that we can use in a deductive proof of safety.

Chapter 6

Liveness verification

Synopsis *In this chapter we (**i**) introduce a necessary condition for eventuality, the existence of what we call a **staging set**, and use it to (**ii**) formulate a rule of inference for formally proving eventuality properties in continuous systems. (**iii**) We illustrate the proof principle using some basic examples and (**iv**) describe how our approach can be used to construct formal proofs of certain liveness properties in a deductive verification tool. Lastly, we (**v**) generalize total derivatives for formulas introduced by Platzer in [Pla10a] by exploiting directional differentiability properties of the* $\min\max$ *function.*

6.1 Introduction

In computer science, by *liveness* one informally understands the property of something "good" happening along the execution paths in a program. Thus, in stating that a program is *live* one asserts that some desirable property will hold true as the program runs. Liveness properties of discrete programs were studied by Lamport and Owicki in [Lam77, OL82] and formally defined by Alpern and Schneider in [AS85]. In this chapter we will be concerned with a particular type of liveness known as *eventuality*, which requires that some *target* set of states is eventually attained. Furthermore, instead of discrete computer programs, we will be working with continuous systems that are governed by ordinary differential equations and have an uncountably infinite number of states.

In this chapter we will develop a new deductive verification method for proving eventuality properties in continuous systems that can be implemented as a rule of inference in a theorem prover for hybrid systems. The method we propose is able to work directly with initial states and target regions given by arbitrary semi-algebraic sets (that is, sets given by finite boolean combinations of polynomial equalities and inequalities) and generalizes previously reported approaches reported in [PR05, RS10, SKA01, Pla10a]. Our approach is not restricted to bounded evolution domains (as e.g. [RS10]) and is able to prove eventuality properties for target regions described by formulas featuring equations (unlike [Pla10a, PR05]). Finally, the presence of system equilibria outside the target region presents an insurmountable obstacle for the approaches reported in [PR05, RS10, Pla10a] and requires the user to manually remove them from the evolution domain [PR05]. We work with weaker conditions that only require a semi-algebraic over-approximation of the reachable set, which can be used to avoid equilibria without the need to manually alter the system. The conditions we give are much more general than in [SKA01] and may be checked automatically using a decision procedure.

6.1.1 Preliminaries

In general, solutions to initial value problems need not be unique or even exist for all time $t \geq 0$, i.e. the maximal interval of existence need not be of the form

(a, ∞). For instance, solutions to simple non-linear systems, such as $\dot{x} = x^2$, already exhibit *finite time blow-up*, i.e. diverge to infinity in finite time. In what follows we will work with differential equations whose solutions are unique and of sufficient duration to allow us to prove properties of interest. For simplicity we sometimes assume that solutions exist for all $t \geq 0$. In such cases, refinements of the arguments are needed if the solutions are of sufficient duration but do not exist for all $t \geq 0$. To remove this problem entirely, it is common (but not necessary) to require the system of ODEs to be Lipschitz continuous, since this property guarantees existence of solutions for all $t \geq 0$. Under the assumption of global existence of solutions, we will refer to the solution φ as the *flow* of the system. In this chapter we will present a direct verification approach that generalizes those previously reported and is at the same time less conservative. To a significant extent, our work will build upon results about invariant sets, which we have discussed in previous chapters. For convenience, below we recall the definition of what constitutes a continuous invariant.

Definition 128. *A semi-algebraic set $I \subseteq \mathbb{R}^n$ is a continuous invariant for $\dot{\vec{x}} = f(\vec{x})$ & H if and only if*

$$\forall\, \vec{x}_0 \in I.\ \forall\, t \geq 0.\ (\forall\, \tau \in [0,t].\ \varphi_\tau(\vec{x}_0) \in H) \rightarrow (\forall\, \tau \in [0,t].\ \varphi_\tau(\vec{x}_0) \in I).$$

We may write a continuous invariance assertion as a formula in $\mathrm{d}\mathcal{L}$ *as follows:*

$$I \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ H]\ I.$$

One useful way of thinking about continuous invariants (this will become apparent later) is to view them as sets that "can only be left by entering $\neg H$ first".

6.2 Direct method for eventuality verification

As a first attempt, one may define eventuality for continuous systems as follows:

$$\forall\, \vec{x}_0 \in \psi.\ \exists\, t \geq 0.\ \big(\varphi_t(\vec{x}_0) \in \mathit{Target}\big),$$

where $\psi \subseteq \mathbb{R}^n$ is the set of initial states and $\mathit{Target} \subseteq \mathbb{R}^n$ is the target set. As with invariants, because the continuous systems we consider may impose

evolution domain constraints $H \subseteq \mathbb{R}^n$, the formal definition of eventuality needs an additional clause stipulating that continuous evolutions remain within the constraint until the target set is attained. Below we give a general definition of eventuality for continuous systems.

Definition 129. *Given a system $\dot{\vec{x}} = f(\vec{x})$ & H, where $H \subseteq \mathbb{R}^n$ is the evolution constraint, $\psi \subseteq H$ is the set of initial states from which solutions are unique and of sufficient duration and Target $\subseteq \mathbb{R}^n$ is the target set of states that we wish the system to attain by starting anywhere inside ψ, then the eventuality property holds if and only if*

$$\forall\, \vec{x}_0 \in \psi.\ \exists\, t \geq 0.\ \big(\, (\forall \tau \in [0,t].\ \varphi_\tau(\vec{x}_0) \in H)\ \wedge\ \varphi_t(\vec{x}_0) \in \mathit{Target}\big),$$

By solutions of sufficient duration we understand solutions that may blow up in finite positive time, but only after reaching Target (finite time blow up in negative time is innocuous for showing eventuality).

We may phrase the eventuality property using a d$\mathcal{L}$ formula as follows:

$$\psi \rightarrow \langle \dot{\vec{x}} = f(\vec{x})\ \&\ H \rangle\ \mathit{Target}.$$

The above formula asserts that if we start anywhere inside ψ, then by following the solution to the system $\dot{\vec{x}} = f(\vec{x})$ & H, we *eventually* (diamond modality $\langle\ \rangle$) reach a state that lies inside *Target*. In using the above formula, we assume that each of the sets H, ψ and *Target* is semi-algebraic and is thus characterized by a quantifier-free formula in the theory of real arithmetic.

Remark 130. In the semantics of d$\mathcal{L}$, the more general statement $\langle \alpha \rangle\ \phi$, where α is a general hybrid program and ϕ is a quantifier-free formula, is true in a state just when there exists some transition in the transition relation of α to a state satisfying formula ϕ (see e.g. [Pla10b, Definition 2.6]). We only work with the special case when α describes a continuous system $\dot{\vec{x}} = f(\vec{x})$ & H, in which case the statement provides a convenient shorthand for expressing eventuality.

Common approaches for proving eventuality properties for continuous systems are often analogous to approaches used in discrete systems, e.g. methods for termination analysis of loops or recursive function calls using well-founded relations as a way of measuring progress. The following example is useful for

exposing some of the intuition as well as the limitations associated with currently existing methods.

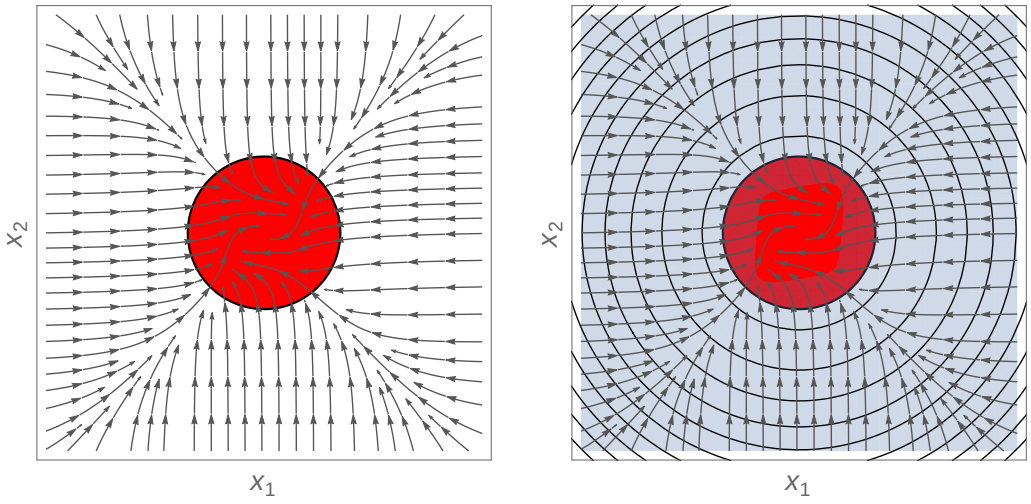
Example 131

Consider the non-linear system

$$\dot{x}_1 = x_2 - 5x_1^5,$$
$$\dot{x}_2 = -x_1^3 - 3x_2^5,$$

with $H = \mathbb{R}^2$ and let the target region be given by

$$\mathit{Target} \equiv x_1^2 + x_2^2 \leq \frac{3}{2}.$$



(a) Target region (in red). (b) Level sets of $p = x_1^2 + x_2^2 - \frac{3}{2}$.

Figure 6.1: Proving eventuality using a decreasing function p. Region where $\mathfrak{L}_f(B) \leq -1$ holds shown in light blue (right). Initial set is any subset of $\mathbb{R}^2$.

In order to show that *Target* is eventually attained from some subset of $\mathbb{R}^2$ we can employ a simple method for showing eventuality where one is required to find a real-valued function p which is *bounded below* and has the property that $\frac{dp(\varphi_t(\vec{x}))}{dt} \leq -\varepsilon$ (equivalently $\mathfrak{L}_f(p) \leq -\varepsilon$, i.e. the value of the function decreases at the rate of at least $-\varepsilon$ along the solutions to the system) is true everywhere inside $\mathbb{R}^2 \setminus \mathit{Target}$ for some fixed positive number ε. For this example, we may simply take $p = x_1^2 + x_2^2 - \frac{3}{2}$ and check that $\mathfrak{L}_f(p) \leq -\varepsilon$ holds true inside $H \setminus \mathit{Target}$ for $\varepsilon = 1$ in order to conclude that *Target* is eventually attained (see Figure 6.1).

Intuitively, this method argues that because p is bounded below, its value cannot be decreasing (at the rate of at least $-\varepsilon$) forever by following the solutions to the

ODE. Therefore, solutions will need to eventually enter a region of H where this requirement is dropped, which is the target region.

As we shall see later in this chapter, duration of solutions is a subtle issue that is soundness-critical and thus requires caution when using this argument. When working with evolution constraints other than the entire space $\mathbb{R}^n$, one is further required to ensure that trajectories cannot leave the evolution constraint H before attaining the target region (this may be ensured by imposing extra conditions on the value of the function p at the closure of the boundary segment of H lying outside the target region; see e.g. [Pra05, Chapter 4]). However, it is also apparent that any system featuring e.g. an equilibrium in H that lies outside the target region cannot be handled using this method because the rate of change of any function along an equilibrium solution is always 0. ■

Generally, by considering the rate of change of the function p everywhere outside the target region (as in Example 131), one is able prove eventuality for *any* initial set in the state space. In practice, when some initial set ψ is supplied, certain regions of the state space may *not* be reachable from ψ and thus not relevant to the eventuality problem at hand. In what follows, we will introduce a much more general proof method to address this limitation.

6.2.1 Staging sets

We now introduce *staging sets*, which are a particular kind of continuous invariants that we use to give an over-approximation of the continuous behaviour in a system with a view to proving eventuality properties without computing solutions to ODEs.

Definition 132. *Given a system $\dot{\vec{x}} = f(\vec{x})$ & H, a set of initial states $\psi \subseteq H$ and a target set of states $Target \subseteq \mathbb{R}^n$, we say that a set $S \subseteq \mathbb{R}^n$ is a* ***staging set*** *if we have $S \subseteq H$, $\psi \setminus Target \subseteq S$ and*

$$\forall\, \vec{x}_0 \in S.\ \forall\, t \geq 0.\ (\forall \tau \in [0,t].\ \varphi_\tau(\vec{x}_0) \notin Target \cap H) \ \rightarrow\ (\forall \tau \in [0,t].\ \varphi_\tau(\vec{x}_0) \in S).$$

One could write this formally using $\mathrm{d}\mathcal{L}$ *as*

$$\big(\psi \wedge \neg Target \rightarrow S\big) \wedge \big(S \rightarrow [\dot{\vec{x}} = f(\vec{x})\ \&\ \neg(Target \wedge H)]\ S\big) \wedge \big(\psi \vee S \rightarrow H\big).$$

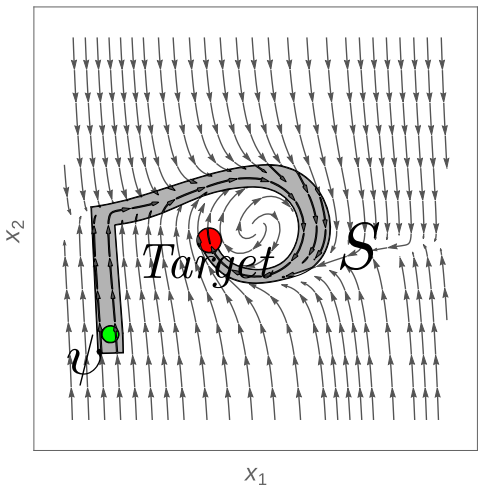


Figure 6.2: Staging set (intuitively). Initial set of states ψ is shown in green, the target set *Target* in red and possible choice for a staging set S in grey; H is taken to be $\mathbb{R}^2$.

Intuitively, a staging set is any set within the evolution constraint H that includes the non-trivial initial states $\psi \setminus \mathit{Target}$ and that "can only be left by entering the region *Target* within the constraint H" or, in other words, provides a "continuous exit window into *Target* within H". Figure 6.2 illustrates this intuition.

In the d$\mathcal{L}$ rendering, it is perhaps surprising to see a clause with a continuous invariant assertion featuring $\neg(\mathit{Target} \wedge H)$ as its evolution constraint. To see this more clearly, compare the condition from the definition of staging set S (top) to that of a continuous invariant I (bottom):

$$\forall\, \vec{x}_0 \in S.\ \forall\, t \geq 0.\ (\forall\, \tau \in [0,t].\ \varphi_\tau(\vec{x}_0) \notin \mathit{Target} \cap H) \quad \rightarrow (\forall \tau \in [0,t].\ \varphi_\tau(\vec{x}_0) \in S),$$
$$\forall\, \vec{x}_0 \in I.\ \forall\, t \geq 0.\ (\forall\, \tau \in [0,t].\ \varphi_\tau(\vec{x}_0) \in H) \quad \rightarrow (\forall\, \tau \in [0,t].\ \varphi_\tau(\vec{x}_0) \in I).$$

The other clauses in the d$\mathcal{L}$ formula ensure that $S \subseteq H$ and $\psi \setminus \mathit{Target} \subseteq S$ hold. Thus, the continuous invariance assertion does not consider trajectories that go outside of H before reaching $\mathit{Target} \cap H$ from any non-trivial initial state.

Let us remark that staging sets are very natural because their existence is a necessary pre-requisite for the eventuality property to hold.

Proposition 133. *If the eventuality property holds for $\dot{\vec{x}} = f(\vec{x})$ & H with initial and target sets $\psi \subseteq H$, $\mathit{Target} \subseteq \mathbb{R}^n$ as before, then there exists a staging set for the system.*

Proof. Assuming the eventuality property holds true in the system, we have $\psi \subseteq H$ and for each $\vec{x}_0 \in \psi \setminus \mathit{Target}$ there exists some $t > 0$ such that $\varphi_t(\vec{x}_0) \in \mathit{Target}$ and $\forall\ \tau \in [0,t].\ \varphi_\tau(\vec{x}_0) \in H$. Now define $\gamma(\vec{x}_0) \equiv \{\varphi_{t'}(\vec{x}_0) \mid t' \in [0,t)\}$ to construct a staging set $S \equiv \bigcup_{\vec{x}_0 \in \psi} \gamma(\vec{x}_0)$. □

Remark 134. The construction in the proof above gives a staging set which may not possess a closed-form description. In practice, by restricting attention to semi-algebraic sets, one can *decide* whether a given candidate set constitutes a staging set for the system at hand. Also, note that if S is a staging set, then $S' \equiv S \setminus \mathit{Target}$ is also a staging set.

Searching for a staging set is in principle no different to searching for a continuous invariant for safety verification. Methods for continuous invariant generation can therefore be applied to search for staging sets. Techniques for continuous invariant generation are still an active area of research (see Chapter 5). Furthermore, certain invariant generation methods may be more suitable for finding staging sets than others. For instance, the sum-of-squares techniques for computing polynomial sub-level set approximations of the finite-time reachable set due to Wang, Lall & West [WLW13] appear promising in this regard.

6.2.2 Progress functions

The existence of a staging set only provides a *necessary condition* for eventuality. In this section we will give a *sufficient condition* that will allow us to soundly conclude the eventuality property. Because we already require the sets we work with to be semi-algebraic, we can invoke a lemma that allows us to conclude eventuality by showing that trajectories leave a given staging set after evolving inside it for some finite amount of time. The context of the lemma is the behaviour of trajectories initialised inside $I \cap H$, where I is some continuous invariant I and H is an evolution constraint, both semi-algebraic.

Before giving the full statement, it is helpful to first consider a continuous function $\vec{x} : [0,a) \to \mathbb{R}^n$ defined on the interval $[0,a)$, where $a > 0$, and any subset I of $\mathbb{R}^n$ such that $\vec{x}(0) \in I$. There are four possibilities to consider:

1. $\vec{x}(t)$ never leaves I, i.e.

$$\forall\, t \in [0, a).\ \vec{x}(t) \in I.$$

2. For some $t_1 \in (0, a)$, $\vec{x}(t)$ is in I while $t < t_1$ but leaves I (at least momentarily) when $t = t_1$, i.e.

$$\exists\, t_1 \in (0, a).\ \forall\, t \in [0, t_1).\ \vec{x}(t) \in I\ \wedge\ \vec{x}(t_1) \notin I.$$

3. For some $t_1 \in [0, a)$ and $t_2 \in (t_1, a)$, $\vec{x}(t)$ is in I while $t \leq t_1$ but then leaves I for some non-trivial time interval (t_1, t_2), i.e.

$$\exists\, t_1 \in [0, a).\, \exists\, t_2 \in (t_1, a).\ ((\forall\, t \in [0, t_1].\ \vec{x}(t) \in I)\ \wedge\ (\forall\, t \in (t_1, t_2).\ \vec{x}(t) \notin I))\,.$$

4. For some $t_1 \in [0, a)$ it is the case that $\vec{x}(t)$ is in I while $t \leq t_1$, but $\vec{x}(t)$ oscillates in and out of I in any non-trivial time interval (t_1, t_2), i.e.

$$\exists\, t_1 \in [0, a).\ \forall\, t_2 \in (t_1, a).\Big((\forall\, t \in [0, t_1].\ \vec{x}(t) \in I) \\ \wedge\ \big(\exists\, t_3, t_4 \in (t_1, t_2).\ \vec{x}(t_3) \in I\ \wedge\ \vec{x}(t_4) \notin I\big)\Big).$$

When $\vec{x} : [0, a) \to \mathbb{R}^n$ is an analytic function on $[0, a)$ and I is a semi-algebraic set, the function $\mathrm{sgn}(\vec{x}(t))$ would be constant on some open interval $(0, b)$, where $0 < b \leq a$. This implies that for any given polynomial p_i defining the semi-algebraic set I, the function $\mathrm{sgn}(p_i(\vec{x}(t_1 + t)))$ would remain constant on some non-empty open time interval $(0, b_i)$, where $0 < b_i < a - t_1$. By taking $b = \min_i b_i$, we see that either $\vec{x}(t_1 + t) \in I$ for all $t \in (0, b)$, or $\vec{x}(t_1 + t) \notin I$ for all $t \in (0, b)$, and so case 4 cannot hold.

Now, under the assumption of analyticity of solutions φ_t and the semi-algebraic nature of the sets I and H, there are three possibilities for a trajectory initialised at $\vec{x}_0 \in I \cap H$:

1. the trajectory may "leave H while still in I", i.e.

$$\exists\, \tau \geq 0.\ (\forall\, t \in [0, \tau].\ \varphi_t(\vec{x}_0) \in I) \wedge \varphi_\tau(\vec{x}_0) \notin H,$$

2. the trajectory may "leave I and H at the same time", i.e.

$$\forall\, \tau \geq 0.\ (\forall\, t \in [0,\tau].\ \varphi_t(\vec{x}_0) \in I) \iff (\forall\, t \in [0,\tau].\ \varphi_t(\vec{x}_0) \in H),$$

3. the trajectory may "leave I while still in H", i.e.

$$\exists\, \tau \geq 0.\ (\forall\, t \in [0,\tau].\ \varphi_t(\vec{x}_0) \in H) \wedge \varphi_\tau(\vec{x}_0) \notin I.$$

Observe that case 3 is impossible if I is a continuous invariant for the system, which leaves us to consider cases 1 and 2 in the lemma given below, with a further detail for case 2 that exists when sets I and H are semi-algebraic.

Lemma 135. *If $H, I \subseteq \mathbb{R}^n$ are semi-algebraic and I is a continuous invariant for the system $\dot{\vec{x}} = f(\vec{x})$ & H then any solution that starts in $I \cap H$ and subsequently leaves I either **(i)** leaves H while still in I or **(ii)** leaves H and I at the same time and has a non-empty segment[1] immediately on leaving I that is wholly contained in $\mathbb{R}^n \setminus H$ (i.e. $\neg H$).*

Remark 136. In cases when sets are not semi-algebraic, the lemma is false. This can be seen from a simple counter-example in which $I \equiv x \leq 0$, H is given by $I \cup \{x \mid \sin(\frac{1}{x}) = 0, x > 0\}$ and $\dot{x} = 1$ defines the dynamics. Even though I and H are left at the same time, there is no non-empty segment of the trajectory immediately on leaving I that is wholly contained in $\neg H$. Note that this behaviour corresponds to the oscillations described earlier in case 4, and cannot occur with semi-algebraic sets and analytic functions.

Proof. Assume case (i) is false, in which case we have that I and H are left at the same time and we have to show property (ii). For (ii) we need to show that if I and H are left at the same time, then immediately afterwards the state of the system satisfies $\neg H$ for some non-empty time interval. If there is a time $t' > 0$ such that $\forall\ \tau \in [0,t').\ \varphi_\tau(\vec{x}_0) \in H \cap I$ and $\varphi_{t'}(\vec{x}_0) \notin H \cup I$, then the state of the system satisfies $\neg H$ for $[t',t']$ immediately upon leaving I. If no such t' exists, consider a point $\vec{x}_1 \in I \cap H$ from which the system can no longer

[1] This includes points.

evolve inside I without violating the constraint H. It is necessarily the case that $\forall\ \epsilon > 0.\ \exists\ t \in (0,\epsilon).\varphi_t(\vec{x}_1) \notin H$ holds, i.e. no further motion of the system can satisfy the evolution constraint. We need to show the stronger property $\exists\ \epsilon > 0.\ \forall\ t \in (0,\epsilon).\varphi_t(\vec{x}_1) \notin H$. For any semi-algebraic set, let $P \subset \mathbb{R}[\vec{x}]$ be the collection of polynomials appearing in its description. At the point $\vec{x}_1$ for each $p_i \in P$ we have that $p_i(\vec{x}_1) \sim 0$, where $\sim \in \{<, =, >\}$. For those $p_i \in P$ such that $p_i(\vec{x}_1) > 0$ or $p_i(\vec{x}_1) < 0$, there is guaranteed to be an open neighbourhood U_i around $\vec{x}_1$ for which $p_i(U_i) > 0$ or $p_i(U_i) < 0$ holds (since polynomials are continuous functions). Therefore, there is some non-empty time neighbourhood $(0,\epsilon)$ for which the motion of the system will satisfy the strict sign conditions. When $p_i(\vec{x}_1) = 0$, one either has $\mathfrak{L}_f^k(p_i(\vec{x}_1)) = 0$ for infinitely many orders k, or there exists an $k \geq 1$ such that $\mathfrak{L}_f^k(p_i(\vec{x}_1)) \neq 0$. Since polynomials and solutions to polynomial ODEs are analytic functions, there is some open time neighbourhood $(0,\epsilon)$ where the sign condition on the polynomial p_i is satisfied under the motion (see e.g. [LZZ11, Proposition 9]). Thus, if the system cannot evolve any further inside a semi-algebraic set, then immediately afterwards it evolves inside the semi-algebraic complement of the set for some non-empty open time interval by following the solution. □

If one can show that any trajectory starting inside a staging set S eventually leaves S, one can use Lemma 135 to conclude the eventuality property. An obvious way of showing that S is eventually left without computing the solution to the system of ODEs is to search for an appropriate function, whose derivative can be used as a measure of "progress in leaving S".

Proposition 137. *Given a staging set S for some polynomial system $\dot{\vec{x}} = f(\vec{x})\ \&\ H$ with initial and target sets $\psi \subseteq H$, Target $\subseteq \mathbb{R}^n$ respectively and whose* solutions are of sufficient duration*, if there exists a continuously differentiable function $P : \mathbb{R}^n \to \mathbb{R}$ such that*

$$\exists\ \varepsilon > 0.\ \forall\ \vec{x} \in S.\quad \mathfrak{L}_f(P(\vec{x})) \leq -\varepsilon\ \wedge\ P(\vec{x}) \geq 0,$$

then, provided the sets are semi-algebraic, the eventuality property holds.

A function P satisfying the conditions in the above proposition is known as a **progress function** for the staging set S.

Proof. Fix a start point $\vec{x}_0 \in \psi \setminus \mathit{Target}$ from which we want to argue there is a finite flow with end point in *Target* and which is fully contained in H. First we show that there is a finite flow from $\vec{x}_0$ with end point outside of S. Assume that the solution with initial condition $\vec{x}_0$ is of sufficient duration such that either **(i)** the trajectory exits S at some point or **(ii)** the trajectory is inside S up to and including at least some time $\tau > P(\vec{x}_0)/\varepsilon$. In case **(ii)**, a simple application of the fundamental theorem of calculus yields

$$\begin{aligned} P(\varphi_\tau(\vec{x}_0)) - P(\varphi_0(\vec{x}_0)) \;=\; \int_0^\tau \frac{d}{dt} P(\varphi_t(\vec{x}_0))\; dt &= \int_0^\tau \mathfrak{L}_f(P(\varphi_t(\vec{x}_0)))\; dt \\ &\leq \int_0^\tau -\varepsilon\; dt \\ &= -\varepsilon\tau. \end{aligned}$$

Given $P(\varphi_0(\vec{x}_0)) = P(\vec{x}_0)$ we have that $P(\varphi_\tau(\vec{x}_0)) < 0$ which is impossible since $P(\vec{x}_0) \geq 0$ for all $\vec{x}_0 \in S$. Hence case **(i)** must hold. Using case **(i)**, we now apply Lemma 135 to the invariance property of the staging set S. We have that either the trajectory reaches $\mathit{Target} \cap H$ within S and the eventuality property obviously holds, or, on exiting S we immediately have a non-empty segment of the trajectory contained in $\mathit{Target} \cap H$ and the eventuality property holds too. □

Example 138 *Solutions of insufficient duration (due to Platzer [Pla10a])*
Consider the system

$$\begin{aligned} \dot{x}_1 &= 1, \\ \dot{x}_2 &= 1 + x_2^2, \end{aligned}$$

where $H = \mathbb{R}^2$. Suppose we want to show that the set characterized by $x_1 \geq 6$ is eventually attained from any point in $\mathbb{R}^2$. By taking a staging set S defined by $x < 6$ and considering the function $P = -x_1 + 6$, we see that $S \to P \geq 0$ is true and

$$\mathfrak{L}_f(P) = -\dot{x}_1 = -1,$$

from which we would conclude that P is a progress function by e.g. taking $\varepsilon = \frac{1}{2}$. However, solutions in this system diverge to infinity in finite time (i.e. exhibit finite time blow up, as depicted in Figure 6.3) and the system thus fails to satisfy the requirement that solutions be of sufficient duration.

■

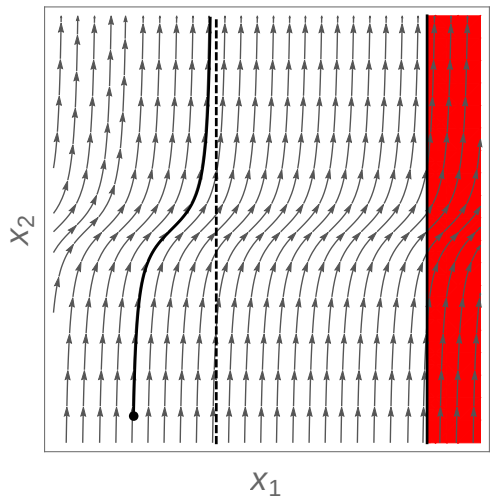


Figure 6.3: Solutions of insufficient duration due to finite time blow up. The trajectory cannot cross the asymptote (dashed) on its way to the target region (in red).

Remark 139. Of course, given some set $\hat{S}$ such that $S \subseteq \hat{S}$, where S is a staging set, if one shows that $\hat{S}$ is left in finite time by following the solutions, then one can also conclude that *Target* is eventually attained. This may seem like a complete waste of effort, but methods developed for *verified integration* of ODEs [BM98, NJN07] can compute *enclosures* of finite-time reachable sets where the enclosure itself is *not* a staging set but is guaranteed to enclose one; in this case, the enclosure can act as $\hat{S}$. Formally verified implementations of enclosure construction algorithms have been reported by Immler [Imm14, Imm15].

Polynomial progress functions may be generated automatically using pre-defined polynomial *templates* of bounded degree with parametric coefficients. The templates can be enumerated (e.g. by successively increasing the polynomial degree) and checked using a real quantifier elimination procedure (such as e.g. CAD [Col75]), leaving the parameters as free variables. The result is a semi-algebraic constraint on the coefficients that will yield a progress function. Of course, the computational complexity of real quantifier elimination [DH88] makes this approach infeasible and therefore practically uninteresting; however, theoretically, one has a *semi-decision procedure* for checking whether a polynomial progress function exists for a given semi-algebraic staging set and a polynomial ODE. Methods based on sum-of-squares techniques (e.g. [PR05]) may offer more practical (albeit incomplete) alternatives for finding progress functions.

6.3 Proof rule for eventuality in ODEs

We are now ready to formalize the proof method for eventuality properties using staging sets and progress functions, as described in the previous section, into a rule of inference.

Proposition 140. *The rule of inference given below (with four premises) is sound with the proviso that solutions are of sufficient duration.*

$$\vdash \exists\, \varepsilon > 0.\ \forall\, \vec{x}.\quad S \to \big(P \geq 0\ \wedge\ \mathfrak{L}_f(P) \leq -\varepsilon\big)$$

$$\text{(SP)}\ \frac{\psi, \neg \mathit{Target} \vdash S \qquad \vdash S \to [\dot{\vec{x}} = f(\vec{x})\ \&\ \neg(H \wedge \mathit{Target})]\ S \qquad \psi \vee S \vdash H}{\vdash \psi \to \langle \dot{\vec{x}} = f(\vec{x})\ \&\ H \rangle\ \mathit{Target}}.$$

Proof. Corollary to 137. The sufficient duration proviso is soundness-critical (see [Pla10a, Counterexample 9] for an example of why this is important). A stronger requirement, e.g. Lipschitz continuity of f (if not globally, then within some compact subset of $\mathbb{R}^n$ containing *Target* and S) may be used to give a formal criterion for ensuring the proviso holds, but this can be restrictive in practice. □

Example 141 *System with limit cycle and equilibrium*
Consider the system of ODEs with an equilibrium and a limit cycle

$$\begin{aligned} \dot{x}_1 &= \ x_2 - x_1\left(x_1^2 + x_2^2 - 1\right), \\ \dot{x}_2 &= -x_1 - x_2\left(x_1^2 + x_2^2 - 1\right), \end{aligned}$$

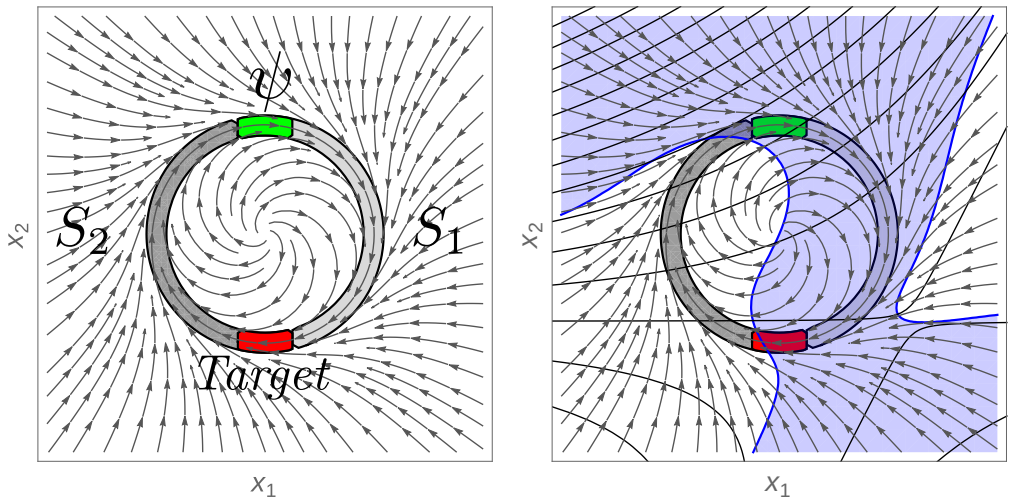
with $H \equiv x_1 \leq 2 \wedge x_1 \geq -2 \wedge x_2 \leq 2 \wedge x_2 \geq -2$ and let the initial set of states and the target region be as follows:

$$\begin{aligned} \psi &\equiv x_2 > 0 \wedge x_1 \geq -\frac{1}{4} \wedge x_1 \leq \frac{1}{4} \wedge (x_1^2 + x_2^2 - 1)^2 \leq \frac{1}{30}, \\ \mathit{Target} &\equiv x_2 < 0 \wedge x_1 \geq -\frac{1}{4} \wedge x_1 \leq \frac{1}{4} \wedge (x_1^2 + x_2^2 - 1)^2 \leq \frac{1}{30}. \end{aligned}$$

Consider also the following sets (depicted in Figure 6.4 on page 161):

$$\begin{aligned} S_1 &\equiv \neg \mathit{Target} \wedge x_1 \geq -\frac{1}{4} \wedge (x_1^2 + x_2^2 - 1)^2 \leq \frac{1}{30}, \\ S_2 &\equiv \neg\psi \wedge x_1 \leq \frac{1}{4} \wedge (x_1^2 + x_2^2 - 1)^2 \leq \frac{1}{30}. \end{aligned}$$

One may check using a decision procedure that S_1 is indeed a staging set for this system.



(a) Initial states ψ (in green), target region *Target* (in red) and staging sets S_1 (in grey and green, i.e. S_1 includes ψ) and S_2 (dark grey and red, i.e. S_2 includes *Target*).

(b) Level sets of the progress function P_1 for showing eventual exit out of S_1 and the region where $\exists\, \varepsilon > 0.\ \mathfrak{L}_f(P_1) \leq -\varepsilon$ holds (includes S_1; shaded in blue).

Figure 6.4: Proving eventuality using a staging set and a progress function.

A possible progress function for S_1 is $P_1(\vec{x}) = -\left(x_1 - \frac{6}{5}\right)^2 + (x_1 - x_2 - 2)^2 + 10$. Computing the total derivative of P_1 (i.e. Lie derivative with respect to the vector field f) we obtain $\mathfrak{L}_f(P_1(\vec{x})) =$

$$2\,(x_1 - x_2 - 2)\left(x_2^3 + x_1^2 x_2 - x_2 + x_1\right) + \frac{2}{5}\,(5x_2 + 4)\left(x_1^3 + \left(x_2^2 - 1\right)x_1 - x_2\right).$$

Using a decision procedure for real arithmetic to check that the sentence

$$\exists\, \varepsilon > 0.\ \forall\, \vec{x} \in S_1. \qquad \mathfrak{L}_f(P_1(\vec{x})) \leq -\varepsilon \ \wedge\ P_1(\vec{x}) \geq 0$$

is true is sufficient to conclude the eventuality property

$$\psi \rightarrow \langle \dot{x}_1 = x_2 - x_1\left(x_1^2 + x_2^2 - 1\right), \dot{x}_2 = -x_1 - x_2\left(x_1^2 + x_2^2 - 1\right)\ \&\ H\rangle\ \mathit{Target}$$

using the proof rule SP with S_1 as the staging set and P_1 acting as the progress function. Similarly, one may instead take *Target* to be the initial set of states and ψ to be the target region. By using S_2 as a staging set and taking the progress function

$$P_2(\vec{x}) = -\left(-x_1 - \frac{6}{5}\right)^2 + (-x_1 + x_2 - 2)^2 + 10$$

one may use the proof rule SP, instantiating S_2 and P_2 appropriately, to prove

$$\mathit{Target} \rightarrow \langle \dot{x}_1 = x_2 - x_1\left(x_1^2 + x_2^2 - 1\right), \dot{x}_2 = -x_1 - x_2\left(x_1^2 + x_2^2 - 1\right)\ \&\ H\rangle\ \psi.$$

When taken together, the two eventuality proofs show that the system oscillates indefinitely between ψ and *Target*. Let us also note that in this example, due to the convergence of solutions to the limit cycle and an equilibrium at the origin, there can be no global progress function for proving the eventuality properties and staging sets are an essential component of the proof. ■

The proof rule SP can be used as part of a formal verification calculus for hybrid systems in which liveness properties of hybrid systems are reduced using rules of inference to proving liveness properties for discrete and continuous sub-components. When working in a proof calculus, the following proof rule, formalizing the transitivity of the eventuality relation between sets of states, is often convenient:

$$(\langle\rangle \text{ Trans}) \ \frac{\vdash \psi \rightarrow \langle \dot{\vec{x}} = f(\vec{x}) \ \& \ H \rangle \ T \qquad \vdash T \rightarrow \langle \dot{\vec{x}} = f(\vec{x}) \ \& \ H \rangle \ \mathit{Target}}{\vdash \psi \rightarrow \langle \dot{\vec{x}} = f(\vec{x}) \ \& \ H \rangle \ \mathit{Target}}.$$

Let us note also that proving the property of *set reachability* reduces to proving the existence of a non-empty set of initial states $R \subseteq \psi$ from which the eventuality property holds. We may formalize this fact in the following proof rule:

$$(\mathit{Reach}) \ \frac{\vdash R \wedge \psi \not\equiv_{\mathbb{R}} \text{False} \quad \vdash R \rightarrow \langle \dot{\vec{x}} = f(\vec{x}) \ \& \ H \rangle \ \mathit{Target}}{\vdash \exists \ \vec{x} \in \psi. \ \langle \dot{\vec{x}} = f(\vec{x}) \ \& \ H \rangle \ \mathit{Target}}.$$

To show that a given set *Target* is eventually attained from some initial set ψ in a hybrid system, one can apply the rule SP to e.g. first show that some *guard set* within a mode is attained and then proceed to compute the sets reachable from the guard set by following the enabled discrete transitions, using these (or their semi-algebraic over-approximation) as the new initial sets in subsequent applications of SP.

The next section will discuss the relationship between SP and an existing proof method called *differential induction* using *differential variants* [Pla10a] that is part of the logic d$\mathcal{L}$ and has been applied to hybrid system liveness verification problems.

6.4 Obtaining progress functions from formulas

In this section we will use directional differentiability properties of the min max functional with differentiable arguments [Dem70, Eki03] to broaden the class of progress functions at our disposal and discuss how this generalizes the definition of total derivative for *formulas* that was used for *differential variants* in [Pla10a]. We will also show how the proof rule SP serves to remove certain limitations inherent in differential variants.

6.4.1 Derivatives of formulas and differential variants

Differential induction using differential variants (and differential invariants) is a direct proof method introduced by Platzer in [Pla10a] for proving eventuality (invariance) properties in ODEs, as part of a verification calculus for hybrid systems. The method allows one to work with arbitrary semi-algebraic sets represented by quantifier-free formulas. Let us recall the definiton of derivation operator D from Chapter 4 (see also [Pla10a, Def. 13]): $D(r) = 0$ for numbers, $D(x) = \dot{x}$ for variables, $D(a+b) = D(a) + D(b)$, where a, b stand for numbers or variables, $D(a \cdot b) = D(a) \cdot b + a \cdot D(b)$ (product rule), $D\left(\frac{a}{b}\right) = \frac{D(a)\cdot b - a\cdot D(b)}{b^2}$ (quotient rule),

$$\begin{aligned} D(F \wedge G) &\equiv D(F) \wedge D(G), && \text{for quantifier-free formulas } F \text{ and } G, \\ D(F \vee G) &\equiv D(F) \wedge D(G), && \wedge \text{ needed for soundness [Pla10a]} \\ D(a \leq b) &\equiv D(a) \leq D(b), && \text{accordingly for } \geq, >, <, = . \end{aligned}$$

The formula $(D(F) \geq \varepsilon)^{f(\vec{x})}_{\dot{\vec{x}}}$ is obtained by applying the derivation operator to formula F, performing a substitution where each $\dot{x}_i$ in $D(F)$ is replaced with the corresponding right-hand side in the differential equation and replacing all inequalities $a \geq b$ by $a \geq b + \varepsilon$ (accordingly for $<, \leq, >$; see [Pla10a, Section 4.6]).

$$\text{(DV)}\ \frac{\vdash \exists\, \varepsilon > 0 (\neg \mathit{Target} \wedge H \rightarrow (D(\mathit{Target}) \geq \varepsilon)^{f(\vec{x})}_{\dot{\vec{x}}})}{[\dot{\vec{x}} = f(\vec{x})\ \&\ {\sim}\mathit{Target}]H \vdash \langle \dot{\vec{x}} = f(\vec{x})\ \&\ H \rangle \mathit{Target}}$$

The formula ${\sim}\mathit{Target}$ is the *weak negation* of *Target* [Pla10a, Section 4.6] defined by the negation of *Target* in which every strict inequality is made non-strict, e.g.

$${\sim}(x_1 > 0 \vee x_2 - x_1 \leq 10 \vee x_2 \leq 1) \equiv x_1 \leq 0 \wedge x_2 - x_1 \geq 10 \wedge x_2 \geq 1.$$

Formulas *Target* provable using the rule DV are called differential variants. Note also that *Target* is required to define a *closed* set for the rule DV to be sound.

Example 142 *Counterexample (due to Jackson)*
Consider the following liveness assertion

$$\langle \dot{x} = 1 \ \& \ x \leq 1 \rangle \ x > 1,$$

which is clearly false as the (open) target region lies outside the evolution constraint. Applying the rule DV, one gets

$$\text{(DV)} \ \frac{\vdash \exists \ \varepsilon > 0. \ (x \leq 1 \wedge x \leq 1 \rightarrow 1 \geq \varepsilon)}{[\dot{x} = 1 \ \& \ x \geq 1] \ x \leq 1 \vdash \langle \dot{x} = 1 \ \& \ x \leq 1 \rangle \ x > 1},$$

where the premise is a true sentence in $\mathbb{R}$, implying that the conclusion holds. An initial state $x = 1$ will satisfy the antecedent of the conclusion, but the formula in the consequent is false. ■

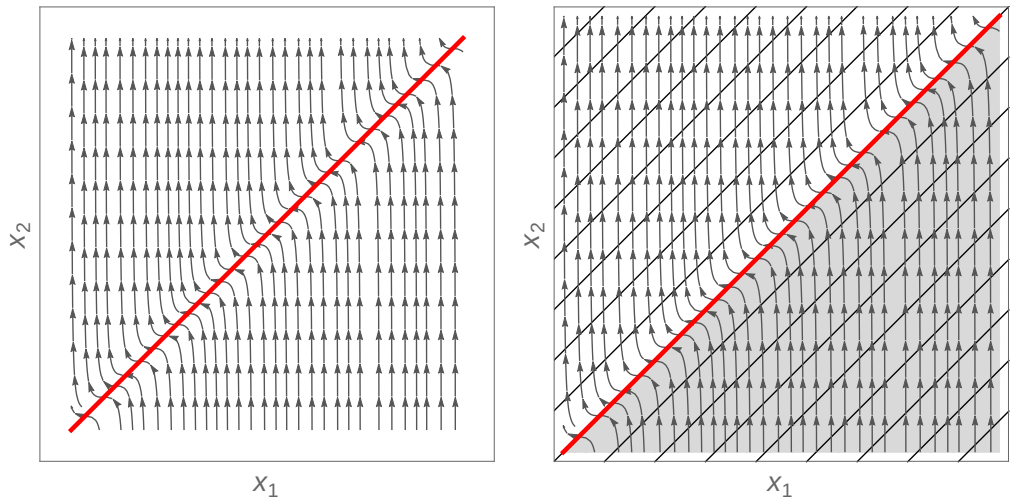
Like our proof rule SP, the rule DV may be applied under the proviso that solutions are of sufficient duration (see [Pla10a, Section 4.7]).

In practice, DV is rather conservative because (apart from not being able to handle systems where H contains equilibria outside the target region) it is incapable of proving eventuality properties for target regions described by equations [Pla10a, Counterexample 7]. In Example 143 we demonstrate a simple proof of such a property using staging sets and progress functions.

Example 143 *Target region with equational description*
Let the dynamics be given by the non-linear system

$$\dot{x}_1 = -1,$$
$$\dot{x}_2 = (x_2 - x_1)^2,$$

$H = \mathbb{R}^2$ and consider a target region described an equation $\mathit{Target} \equiv x_2 - x_1 = 0$ (see Figure 6.5 on page 165).



(a) Target region $\mathit{Target} \equiv x_2 - x_1 = 0$ (in red) and any initial set anywhere below the red line (not shown).

(b) Staging set $S \equiv x_2 - x_1 < 0$ (in grey) and level sets of the progress function $P(\vec{x}) = -(x_2 - x_1)$.

Figure 6.5: Proving eventuality to a target region described by an equational formula.

Suppose the initial set of states ψ is any subset of $\{\vec{x} \in \mathbb{R}^2 \mid x_2 - x_1 < 0\}$. To show the eventuality property let us take $S \equiv x_2 - x_1 < 0$, which can be easily shown to be a staging set, and use $P(\vec{x}) = -(x_2 - x_1)$ as a progress function. The total derivative of P is given by $\mathfrak{L}_f(P(\vec{x})) = -(x_2 - x_1)^2 - 1$, which satisfies the ε-progress property inside the staging set S. An application of the rule SP proves the property $\psi \to \langle \dot{x}_1 = -1, \dot{x}_2 = (x_2 - x_1)^2 \;\&\; H \rangle\ \mathit{Target}$. ■

In general, finding an appropriate progress function P for use with the rule SP can be rather non-trivial; however, sometimes the description of the target region itself may suggest a progress function. Indeed, this is how the rule DV checks the ε-progress property towards the target region: by considering the total derivative of the formula giving the target region itself. This is not guaranteed to work even if the eventuality property is true, but one may think of DV as generating a "progress formula" from the description of the target region. Because DV relies on the derivation operator D for its notion of ε-progress for formulas, the resulting conditions are very strong. In what follows, we will seek to relax them, while still using the description of the target region to suggest a progress function that can be used with our proof method.

6.4.2 Non-differentiable progress functions

Given a quantifier-free formula *Target* characterizing a semi-algebraic set, the weak negation of its negation, $\sim\neg$*Target* ($\sim$ defined as for DV), gives a formula in which every strict inequality symbol of *Target* is made non-strict. The resulting formula characterizes a *closed* semi-algebraic set that over-approximates the closure of *Target*.

Remark 144. To see why this over-approximation may not give the exact characterization of the closure, consider the formula $(x_1^2 + x_2^2 - 1)(x_1^2 + x_2^2) > 0$. Geometrically, this set represents the complement in $\mathbb{R}^2$ of a closed unit disc centred at the origin. By making the strict inequality non-strict we obtain $(x_1^2 + x_2^2 - 1)(x_1^2 + x_2^2) \geq 0$, which defines a region that includes the origin, which is not a boundary point of the original set.

Note that any closed semi-algebraic set can always be put into the form

$$\bigvee_{i=1}^{n} \bigwedge_{j=1}^{m(i)} p_{ij} \leq 0,$$

where p_{ij} are polynomials. The set of states satisfying such a formula can equivalently be expressed as a sub-level set of a continuous function, i.e.

$$\min_{i\in[1,n]} \max_{j\in[1,m(i)]} p_{ij} \leq 0.$$

Although this function need not be differentiable, for ensuring the property of ε-progress, viz. $\mathfrak{L}_f(\cdot) \leq -\varepsilon$, we are merely interested in a certain condition on its *directional derivative* in the direction of the vector field f. Directional differentiability properties of the min max function have previously been investigated in non-smooth analysis [Eki03, Dem70] and it was shown that under certain mild assumptions (see [Eki03]), the min max function has a directional derivative that can also be expressed as a min max function. Furthermore, these assumptions are guaranteed to hold if the ε-progress property is satisfied. The directional derivative of min max (see [Eki03]) in the direction of the vector field f, may be used to define

$$\mathfrak{L}_f(\min_{i\in[1,n]} \max_{j\in[1,m(i)]} p_{ij}) = \min_{i\in I_*} \max_{j\in J_*} (\mathfrak{L}_f(p_{ij}))\,,$$

where p_{ij} are differentiable real-valued functions and

$$J_* = \{j_* \in [1, m(i)] \mid p_{ij_*} = \max_{j \in [1,m(i)]}(p_{ij})\},$$
$$I_* = \{i_* \in [1, n] \mid p_{i_* j} = \min_{i \in [1,n]} \max_{j \in [1,m(i)]}(p_{ij})\}.$$

The above definition may at first sight appear rather opaque; the following illustrative example is useful in exposing some of the intuition.

Example 145

Suppose that we have a formula $F \equiv p_1 \leq 0 \wedge p_2 \leq 0$. Then we have $F \equiv_{\mathbb{R}} \max(p_1, p_2) \leq 0$ and the directional derivative along f given by

$$\mathfrak{L}_f \max(p_1, p_2) = \begin{cases} \mathfrak{L}_f(p_1) & p_1 > p_2 \\ \mathfrak{L}_f(p_2) & p_2 > p_1 \\ \max(\mathfrak{L}_f(p_1), \mathfrak{L}_f(p_2)) & p_1 = p_2 \end{cases}$$

Intuitively, when there is only one differentiable "active component" (i.e. a function p_j which evaluates to the same value as the whole max function), the directional derivative is simply given by $\mathfrak{L}_f(p_j)$; however, when there are many, the index set J_* contains more than one element and the directional derivative is given by $\max_{j \in J_*} \mathfrak{L}_f(p_j)$ where all p_j are currently active. More generally, once the directional derivative of $\min \max p_{ij}$ is computed and an ε-progress condition is imposed, the resulting expression will feature conditionals involving min, max, ε and p_{ij}s and can thus be converted back into a formula giving precisely the conditions for the ε-progress of the min max function. The resulting formulas will often be long and unwieldy, but for this simple example we can write the condition in full:

$$\begin{aligned} \mathfrak{L}_f \max(p_1, p_2) \leq -\varepsilon \equiv\ & \big(p_1 > p_2 \to \mathfrak{L}_f(p_1) \leq -\varepsilon\big) \\ \wedge\ & \big(p_2 > p_1 \to \mathfrak{L}_f(p_2) \leq -\varepsilon\big) \\ \wedge\ & \big(p_1 = p_2 \to \\ & \quad (\mathfrak{L}_f(p_1) \geq \mathfrak{L}_f(p_2) \to \mathfrak{L}_f(p_1) \leq -\varepsilon) \wedge \\ & \quad (\mathfrak{L}_f(p_1) < \mathfrak{L}_f(p_2) \to \mathfrak{L}_f(p_2) \leq -\varepsilon)\ \big). \end{aligned}$$

Similarly, if one wanted to impose the ε-progress property towards the formula $F \equiv p_1 \leq 0 \vee p_2 \leq 0$, encoded as $F \equiv_{\mathbb{R}} \min(p_1, p_2) \leq 0$, one would obtain

$$\begin{aligned}\mathfrak{L}_f \min(p_1, p_2) \leq -\varepsilon \equiv \big(&p_1 < p_2 \rightarrow \mathfrak{L}_f(p_1) \leq -\varepsilon\big) \\ \wedge\ \big(&p_2 < p_1 \rightarrow \mathfrak{L}_f(p_2) \leq -\varepsilon\big) \\ \wedge\ \big(&p_1 = p_2 \rightarrow \\ &(\mathfrak{L}_f(p_1) \leq \mathfrak{L}_f(p_2) \rightarrow \mathfrak{L}_f(p_1) \leq -\varepsilon) \wedge \\ &(\mathfrak{L}_f(p_1) > \mathfrak{L}_f(p_2) \rightarrow \mathfrak{L}_f(p_2) \leq -\varepsilon)\ \big).\end{aligned}$$

By nesting these definitions appropriately, using facts such as e.g. $\min(p_1, p_2, p_3) = \min(p_1, \min(p_2, p_3))$, one can arrive at ε-progress conditions for more complicated closed semi-algebraic sets. ■

Remark 146. Similar tools and ideas have been employed in sufficient conditions for positive invariance of certain sets with non-smooth boundaries (e.g. *practical sets* in [BM08] and closed semi-algebraic sets [TT09]). These approaches are based on Nagumo's theorem [Nag42] and require computing/under-approximating the *contingent cone*, which can be defined in terms of limits of directional derivatives. The interested reader is invited to consult [Eki03] for a more detailed exposition of the technical assumptions used in formulating the directional derivative of min max.

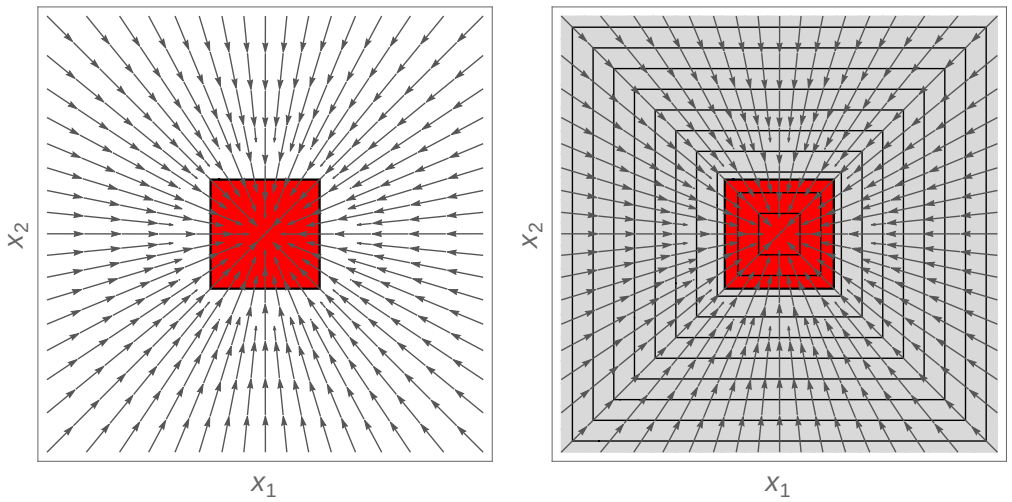
Example 147 *Non-differentiable progress function*
Consider the continuous system

$$\begin{aligned}\dot{x}_1 &= -x_1, \\ \dot{x}_2 &= -x_2,\end{aligned}$$

$H = \mathbb{R}^2$ and let the target set of states correspond to a 2×2 box centred at the origin, i.e. $\mathit{Target} \equiv x_1 \leq 1 \wedge x_1 \geq -1 \wedge x_2 \leq 1 \wedge x_2 \geq -1$. From the phase portrait in Figure 6.6 on page 169 (left) it is clear that the eventuality property is true, i.e. by starting the system outside the box, we are guaranteed to eventually enter the box by following the flow.

This property cannot be proved directly using the rule DV because the definition of the derivation operator for formulas requires one to show that *each* conjunct is a differential variant. In this case,

$$D(\mathit{Target}) \geq \varepsilon \equiv \dot{x}_1 \leq \varepsilon \wedge \dot{x}_1 \geq \varepsilon \wedge \dot{x}_2 \leq \varepsilon \wedge \dot{x}_2 \geq \varepsilon.$$



(a) Phase portrait and target region *Target* (in red).

(b) Level curves of the max function (black) and a staging set $S \equiv \neg \mathit{Target}$ (grey).

Figure 6.6: Proof of eventuality using a non-differentiable progress function.

Upon substituting the dynamics, this leads to unsatisfiable conditions (since $\varepsilon > 0$):

$$(D(\mathit{Target}) \geq \varepsilon)^{f(\vec{x})}_{\dot{\vec{x}}} \equiv -x_1 \leq \varepsilon \wedge -x_1 \geq \varepsilon \wedge -x_2 \leq \varepsilon \wedge -x_2 \geq \varepsilon \equiv_{\mathbb{R}} \text{False}.$$

Instead, one may write down the formula for the box as a sub-level set, i.e.

$$\mathit{Target} \equiv \max(x_1 - 1, -x_1 - 1, x_2 - 1, -x_2 - 1) \leq 0$$

and taking the complement of *Target* to be the staging set, i.e. $S \equiv \neg \mathit{Target}$, check that

$$\begin{aligned} \exists\, \varepsilon > 0.\ \forall \vec{x} \in S.\big(&\max(x_1 - 1, -x_1 - 1, x_2 - 1, -x_2 - 1) \geq 0 \\ &\wedge\ \mathfrak{L}_f \max(x_1 - 1, -x_1 - 1, x_2 - 1, -x_2 - 1) \leq -\varepsilon\big) \end{aligned}$$

is valid, which is sufficient to conclude the eventuality property for any $\psi \subseteq S$. ■

6.5 Related work

Prajna and Rantzer investigated automatic verification of eventuality properties for ODEs in [PR05]; their approach ensures that evolution occurs within

the domain constraint by imposing extra constraints on the function used to demonstrate progress along the solutions. Furthermore, the ε-progress property is required to hold everywhere outside the target region. System equilibria lying outside the target region present a problem for this approach and need to be manually removed from the evolution domain. Ratschan and She introduced set-Lyapunov functions to study attraction to target regions in [RS10], considering only bounded domains and also imposing conditions for ensuring progress along the solutions everywhere outside the target region, which suffers from the same problem. The proof method we have proposed works with a more general class of eventuality verification problems (as it makes fewer assumptions about the problem statement and the nature of the system) and can handle systems with equilibria outside the target region by appropriately over-approximating the reachable set using staging sets. Our approach is fundamentally different from that used by Platzer in [Pla10a], e.g. allowing target regions with equational descriptions (among other things; see Section 5). Ideas broadly similar to staging sets were explored by Stiver et al. in [SKA01] using *common flow regions*. Informally, common flow regions are sets bounded by invariant manifolds and an "exit boundary". The conditions given in [SKA01] require the target and the common flow regions to be given by a conjunction of sub-level sets of smooth functions and the defining polynomials (except the exit boundary) to be conserved quantities of the system. Conditions for staging sets are more general and less conservative. Lastly, unlike previous approaches, we completely decouple the progress property (using progress functions) from conditions for over-approximating the reachable set of the system (using staging sets).

6.6 Summary

We have developed a very general proof principle for eventuality properties of continuous systems governed by polynomial ODEs under semi-algebraic evolution constraints that works without computing the solutions and can be shown to both extend and generalize previous approaches in [PR05, RS10, Pla10a, SKA01]. We have presented a formalization of our method in a proof rule (SP) which is very well suited for use as part of a formal verification calculus for hybrid systems.

Our work addressed some important theoretical limitations inherent in available methods for eventuality verification; however, much future work remains before scalable formal verification tools can emerge and be applied in practice to large, industrially relevant verification problems. The two most important practical obstacles are the current dearth of scalable methods for continuous invariant (staging set) generation and limited tool support for searching for progress functions. Searching for staging sets is no different to generating continuous invariants, so improved invariant generation tools developed for safety verification of continuous systems can be applied to search for staging sets. Automatically generating progress functions is likewise a difficult problem and would greatly benefit from improved tools for non-linear optimization. We should note that these problems are pervasive in direct methods and are not limited to safety and liveness verification. In the control and dynamical systems community, direct methods for proving the property of *stability* [Lya92] are considered standard, but do not provide the means of computing the stability-proving (Lyapunov) function; this task is delegated to the user and is the focus of much ongoing work to facilitate their automatic discovery (see e.g. [Par00]).

Chapter 7

Conclusion and future work

The deductive approach to hybrid system verification is fraught with many difficulties. Perhaps among the most challenging ones are problems associated with the continuous fragment, which are particularly prominent in models of hybrid systems studied in control theory.

Our thesis focused on addressing some fundamental obstacles that currently impede progress in deductive verification of systems where continuous state evolution is modelled using non-linear ordinary differential equations. In order to work with continuous dynamics described using non-linear ODEs, deductive verification tools crucially depend on the ability to reason about safety and liveness properties without solving the initial value problem. Below we give a short summary of our contributions to furthering this goal.

Summary of contributions

- In Chapter 4 we presented a detailed survey of the available methods for directly checking continuous invariants, which are crucial for deductive safety verification. We have identified soundness issues that can arise when using some of the methods and described techniques that can be employed to simplify the verification problem in a proof calculus and extended some invariant checking methods to handle larger classes of invariants. To our knowledge, this is the first such survey of direct invariant checking methods.

- In Chapter 5 we presented a method for computing exact discrete semi-algebraic abstractions of polynomial continuous systems based on a decision procedure for semi-algebraic continuous invariant checking [LZZ11]. We have identified a soundness issue in an existing method for computing discrete semi-algebraic abstractions [TK02, Tiw08a] and have shown how this problem is eliminated using our abstraction method. The abstractions computed using our method do not suffer from coarseness as they do not feature transitions between discrete states that are impossible in the concrete continuous system.

- Also in Chapter 5, we introduced algorithms for automatic continuous invariant generation that combine exact semi-algebraic abstraction of polynomial continuous systems with sound proof rules for safety verification to improve their scalability. The continuous invariants generated using our algorithms often possess intricate boolean structure that cannot realistically be obtained using template-based invariant generation methods. We collected a set of 100 continuous safety verification problems featuring interesting non-linear systems (see Appendix A) and applied our invariant generation algorithms to these problems in order to empirically assess their performance. The results we observe are highly encouraging and we are confident that our invariant generation algorithms will greatly help the task of proof automation in deductive verification frameworks for hybrid systems with non-linear continuous dynamics.

- In Chapter 6 we introduced a new direct proof method for eventuality properties that is much more powerful than methods available previously. The method provides conditions for eventuality that can be checked using a combination of a decision procedure for semi-algebraic invariant checking and a decision procedure for real arithmetic, making it highly amenable to use in a theorem proving environment. Unlike methods based on constructing enclosures for solutions to ODEs, which require initial sets to be compact and connected, our method is able to work with general semi-algebraic initial regions.

7.1 Limitations and challenges

Poor scalability of decision procedures for real arithmetic (as discussed in Section 2.3) is likely to remain a key obstacle for exact symbolic analysis and verification of continuous and hybrid systems. In practice, we see that real arithmetic problems arising from verification of temporal properties in systems with more than 2 state variables are often beyond the capabilities of currently available decision procedures. However, this fact does not remove the need for sound reasoning principles that are able to work with exact symbolic set representations. By working with certain restricted classes of sets, one may seek to avoid some of the complexity associated with real arithmetic at the price of losing the expressiveness afforded by semi-algebraic sets. Furthermore, it is possible that real arithmetic problems generated during the verification of real engineering systems possess certain structure that may be exploited. It is an interesting question whether decision procedures for real arithmetic can be tailored to perform well on problems that arise in practice. To address this question would require a very large corpus of problems before any meaningful analysis of their structure can be performed. Customized implementations of decision procedures for the existential fragment real arithmetic were previously investigated by Passmore [Pas11] (RAHD), Jovanović and de Moura [JdM12] (nlsat).

Transcendental functions appearing in ordinary differential equations present another serious challenge to formal verification that is only now starting to be addressed. Liu et al. [LZZZ15] reported a procedure for converting systems of ODEs with elementary functions to systems of polynomial ODEs using a technique known as *re-casting*, which was previously employed as a useful heuristic by numerous authors [Pow59, SV87, PP05, Pla10a]. This allows one to apply the verification methods developed for polynomial systems; however, the re-casting step comes at the price of introducing new state variables. Alternatively, one may opt to work with non-polynomial systems without first re-casting them. We explored this approach in our work [JSBP14] on the integration of MetiTarski [AP10], an automatic theorem prover for first-order sentences involving *inequalities* with polynomials and elementary functions, and the KeYmaera interactive theorem prover for hybrid systems [PQ08]. While decidability is lost once elementary functions are allowed in first-order sentences

and one may no longer use the decision procedure for semi-algebraic invariant checking [LZZ11], it is not clear whether it is more practical to work with non-polynomial systems directly (using e.g. sufficient conditions for invariant checking from Chapter 4 that are able to work with elementary functions) or re-cast them to larger polynomial ODEs and apply verification techniques developed for polynomial systems.

A very important problem standing in the way of improved verification tools for continuous and hybrid dynamical systems is the lack of verification benchmarks that can claim to be representative of the problems that are encountered in practice. In Appendix A we have collected 100 safety verification problems for continuous systems with a focus on non-linearity of the ODEs. Most of the problems have been crafted from ODEs found in textbooks on the qualitative theory of non-linear systems; some were taken directly from papers on safety verification of hybrid systems, where interesting non-linear continuous dynamics is still rarely encountered. As more verification tools for hybrid systems are developed, the need for a set of verification benchmarks that can be used to compare these tools will only increase (e.g. see [BBJ15] for recent efforts in this direction).

7.1.1 Future work

We envisage fruitful future work in combining model checking-like verification approaches based on sound discrete abstraction and reachable set enclosure construction, as briefly outlined in Section 5.7 of Chapter 5. We feel that a standalone unbounded time safety verification tool that is able to work with non-linear continuous dynamics would be a valuable addition to the array of verification tools that are available presently.

Further investigation of the various sources of polynomials used in the discretisation step when performing discrete semi-algebraic abstraction can potentially lead to improved verification tools based on model checking. Any progress in this area will also lead to improved performance in our invariant generation algorithms, as they crucially rely on the quality of the underlying discrete abstractions. Our work on computing exact semi-algebraic abstractions

of polynomial continuous systems in Chapter 5 can be further extended to work with hybrid dynamical systems, following the work of Tiwari [Tiw08a], which will require handling transition guards and reset maps in addition to multiple states with continuous evolution.

There are exciting opportunities for extending our work on liveness (Chapter 6) by creating automatic procedures to search for staging sets and progress functions. In the future we hope to augment our proof method with heuristics that will aid mechanised proof construction inside a proof calculus (i.e. proof procedures). We are also hopeful that our technique will see adoption as a proof rule in a theorem prover for hybrid systems. We have already implemented the method of non-smooth strict barrier certificates (Theorem 92 in Chapter 4), as well as the decision procedure for semi-algebraic continuous invariant checking [LZZ11] to work with the KeYmaera prover. However, the implementation of KeYmaera based on the KeY prover is currently being phased out in favour of a completely new system called KeYmaeraX [FMQ+15]. We feel that implementing the methods developed in this thesis as rules of inference for the new KeYmaera system will certainly be desirable. We leave this for future work. Once implementations become readily available, it will also become necessary to experiment with real engineering applications in order to refine and automate the proof methods.

7.1.2 Final remarks

The challenge posed by the formal qualitative analysis of non-linear ODEs is an enormous one and is certain to witness intensive research in many branches of science for many years and decades to come. At present, relatively few groups working in formal verification in the context of hybrid dynamical systems have considered systems with non-linear continuous dynamics, largely due to the difficulties associated with their analysis. We regard our thesis as putting a step towards a very ambitious goal of making non-linear ODEs supported and handled effectively by formal verification tools for hybrid dynamical systems.

Appendix A

Safety verification problems

1. Van der Pol system ($\mu = 1$).

$$\begin{aligned}
\dot{x_1} &= x_2, \\
\dot{x_2} &= (1 - x_1^2)x_2 - x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv (x_1 = 0 \wedge x_2 = 1), \\
\phi &\equiv \left(x_1^2 + x_2^2 \leq 10\right).
\end{aligned}$$

2. Van der Pol system ($\mu = 1$) constrained to the fourth quadrant.

$$\begin{aligned}
\dot{x_1} &= x_2, \\
\dot{x_2} &= \left(1 - x_1^2\right) x_2 - x_1, \\
H &\equiv (x_1 \leq 0 \wedge x_2 \geq 0), \\
\psi &\equiv (x_1 = 0 \wedge x_2 = 1), \\
\phi &\equiv \left(x_1^2 + x_2^2 \leq 10\right).
\end{aligned}$$

3. Lotka–Volterra system ($a = 1, b = 1, c = 1, d = 1$) in the first quadrant.

$$\begin{aligned}
\dot{x_1} &= x_1 (1 - x_2), \\
\dot{x_2} &= -(1 - x_1) x_2, \\
H &\equiv (x_1 \geq 0 \wedge x_2 \geq 0), \\
\psi &\equiv (x_1 = 0 \wedge x_2 = 1), \\
\phi &\equiv (x_2 \geq 0).
\end{aligned}$$

4. Unstable closed trajectory on unit circle.

$$\begin{aligned}
\dot{x_1} &= x_1^3 + x_2^2 x_1 - x_1 - x_2, \\
\dot{x_2} &= x_2^3 + x_1^2 x_2 - x_2 + x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 = 0 \wedge x_2 = \frac{1}{2} \right), \\
\phi &\equiv \left(x_1^2 + x_2^2 \leq 1 \right).
\end{aligned}$$

5. Unstable closed trajectory on unit circle.

$$\begin{aligned}
\dot{x_1} &= x_1^3 + x_2^2 x_1 - x_1 - x_2, \\
\dot{x_2} &= x_2^3 + x_1^2 x_2 - x_2 + x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 = \frac{1}{2} \wedge x_2 = \frac{1}{2} \right), \\
\phi &\equiv \left(x_1^2 + x_2^2 \leq 2 \right).
\end{aligned}$$

6. Unstable closed trajectory on unit circle in the first quadrant.

$$\begin{aligned}
\dot{x_1} &= x_1^3 + x_2^2 x_1 - x_1 - x_2, \\
\dot{x_2} &= x_2^3 + x_1^2 x_2 - x_2 + x_1, \\
H &\equiv \left(x_1 \geq 0 \wedge x_2 \geq 0 \right), \\
\psi &\equiv \left(x_1 = 0 \wedge x_2 = \frac{1}{2} \right), \\
\phi &\equiv \left(x_1^2 + x_2^2 \leq 1 \right).
\end{aligned}$$

7. Synthetic example.

$$\begin{aligned}
\dot{x_1} &= x_1^2 + x_2^2 - 1, \\
\dot{x_2} &= x_1^2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 = 0 \wedge x_2 = 0 \right), \\
\phi &\equiv \left(x_2 \geq 0 \right).
\end{aligned}$$

8. Arrowsmith & Place [AP92] Fig. 1.29 p. 14.

$$\begin{aligned}
\dot{x_1} &= x_1, \\
\dot{x_2} &= x_2^2, \\
H &\equiv \text{True}, \\
\psi &\equiv (x_1 = 0 \wedge x_2 = -1), \\
\phi &\equiv (x_2 \leq 0).
\end{aligned}$$

9. Arrowsmith & Place [AP92] Fig. 1.30 p. 14.

$$\begin{aligned}
\dot{x_1} &= x_2^2, \\
\dot{x_2} &= x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv (x_1 = 1 \wedge x_2 = 0), \\
\phi &\equiv (x_1 \geq 0).
\end{aligned}$$

10. Arrowsmith & Place [AP92] Fig. 1.31 p. 14.

$$\begin{aligned}
\dot{x_1} &= x_1^2, \\
\dot{x_2} &= (2x_1 - x_2)\, x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv (x_1 = -1 \wedge x_2 = 1), \\
\phi &\equiv (x_1 \leq 0 \wedge x_2 > 0).
\end{aligned}$$

11. Arrowsmith & Place [AP92] Fig. 1.32 p. 14.

$$\begin{aligned}
\dot{x_1} &= -x_1 x_2, \\
\dot{x_2} &= x_1^2 + x_2^2, \\
H &\equiv \text{True}, \\
\psi &\equiv (x_1 = -1 \wedge x_2 = 1), \\
\phi &\equiv (x_1 \leq 0 \wedge x_2 > 0).
\end{aligned}$$

12. Arrowsmith & Place [AP92] Fig. 1.35 p. 17.

$$\begin{aligned}
\dot{x_1} &= x_1 \left(-x_1 - 2x_2 + 2\right), \\
\dot{x_2} &= \left(-2x_1 - x_2 + 2\right) x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 = 1 \wedge x_2 = 1\right), \\
\phi &\equiv \left(x_1 > 0 \wedge x_2 > 0\right).
\end{aligned}$$

13. Arrowsmith & Place [AP92] 3.1 p. 72.

$$\begin{aligned}
\dot{x_1} &= -\left(x_1^2 + x_2^2\right) x_2 - 4x_2 + x_1 \left(-x_1^2 - x_2^2 + 1\right), \\
\dot{x_2} &= \left(x_1^2 + x_2^2\right) x_1 + 4x_1 + x_2 \left(-x_1^2 - x_2^2 + 1\right), \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1^2 + x_2^2 \leq \frac{1}{5}\right), \\
\phi &\equiv \left(x_1^2 + x_2^2 \leq 1\right).
\end{aligned}$$

14. Arrowsmith & Place [AP92] Fig. 3.5(c) p. 79.

$$\begin{aligned}
\dot{x_1} &= x_1 - x_2^3, \\
\dot{x_2} &= x_1^3 + x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1^2 + x_2^2 \geq 1\right), \\
\phi &\equiv \left(x_1^2 + x_2^2 \geq \frac{1}{2}\right).
\end{aligned}$$

15. Arrowsmith & Place [AP92] Fig. 3.5(e) p. 79.

$$\begin{aligned}
\dot{x_1} &= x_1^2 + \frac{1}{2}\left(x_1 + x_2\right), \\
\dot{x_2} &= \frac{1}{2}\left(3x_2 - x_1\right), \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 = 1 \wedge x_2 = -1\right), \\
\phi &\equiv \left(x_2 \leq 0\right).
\end{aligned}$$

16. Arrowsmith & Place [AP92] Fig. 3.8 p. 82.

$$\begin{aligned}
\dot{x_1} &= x_1 \left(x_1 + 2x_2\right), \\
\dot{x_2} &= x_2 \left(2x_1 + x_2\right), \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 = 1 \wedge x_2 = -1\right), \\
\phi &\equiv \left(x_2 \leq 0 \wedge x_1 \geq 0\right).
\end{aligned}$$

17. Arrowsmith & Place [AP92] Fig. 3.9 p. 83.

$$\begin{aligned}
\dot{x_1} &= x_1 \left(x_1 - 2x_2\right), \\
\dot{x_2} &= -\left(2x_1 - x_2\right) x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 = -1 \wedge x_2 = -1\right), \\
\phi &\equiv \left(x_2 \leq 0 \wedge x_1 \leq 0\right).
\end{aligned}$$

18. Arrowsmith & Place [AP92] Fig. 3.10 p. 83.

$$\begin{aligned}
\dot{x_1} &= -x_2^5, \\
\dot{x_2} &= x_2^2 + x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 = -1 \wedge x_2 = 0\right), \\
\phi &\equiv \left(x_1 \leq 10\right).
\end{aligned}$$

19. Arrowsmith & Place [AP92] Fig. 3.11 p. 83.

$$\begin{aligned}
\dot{x_1} &= x_1 - x_2^2, \\
\dot{x_2} &= x_2 \left(x_1 - x_2^2\right), \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 = 1 \wedge x_2 = \frac{1}{8}\right), \\
\phi &\equiv \left(x_1 \geq 0\right).
\end{aligned}$$

20. Vinograd system, Chicone [Chi06] ex. 1.145, p. 80.

$$\begin{aligned}
\dot{x_1} &= x_2^5 + x_1^2 (x_2 - x_1), \\
\dot{x_2} &= x_2^2 (x_2 - 2x_1), \\
H &\equiv \text{True}, \\
\psi &\equiv (x_1 = 1 \wedge x_2 = 0), \\
\phi &\equiv (x_2 = 0).
\end{aligned}$$

21. Duffing equation ($\delta = 1, \beta = -1, \alpha = 1$).

$$\begin{aligned}
\dot{x_1} &= x_2, \\
\dot{x_2} &= -x_1^3 + x_1 - x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left((x_1 + 1)^2 + x_2^2 \leq \frac{1}{4} \right), \\
\phi &\equiv (x_1 < 1 \wedge (x_1 \neq 0 \vee x_2 \neq 0)).
\end{aligned}$$

22. Duffing equation ($\delta = 0, \beta = -1, \alpha = 1$).

$$\begin{aligned}
\dot{x_1} &= x_2, \\
\dot{x_2} &= x_1 - x_1^3, \\
H &\equiv \text{True}, \\
\psi &\equiv \left((x_1 + 1)^2 + x_2^2 \leq \frac{1}{4} \right), \\
\phi &\equiv (x_2 < 3).
\end{aligned}$$

23. Hamiltonian system.

$$\begin{aligned}
\dot{x_1} &= -2x_2, \\
\dot{x_2} &= -3x_1^2 - 2x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 + \frac{2}{3} \right)^2 + x_2^2 \leq \frac{1}{24} \right), \\
\phi &\equiv (x_1 \leq 0).
\end{aligned}$$

24. Hamiltonian system generated from the Motzkin polynomial.

$$\begin{aligned}
\dot{x_1} &= 2x_2x_1^4 + 4x_2^3x_1^2 - 6x_2x_1^2, \\
\dot{x_2} &= -2x_1x_2^4 - 4x_1^3x_2^2 + 6x_1x_2^2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1^2 + x_2^2 \leq \frac{1}{4}\right), \\
\phi &\equiv \left(x_1^2 + x_2^2 \leq 4\right).
\end{aligned}$$

25. Darboux–Christoffel Integrals, Goriely [Gor01] p. 58.

$$\begin{aligned}
\dot{x_1} &= 3\left(x_1^2 - 4\right), \\
\dot{x_2} &= -x_2^2 + x_1x_2 + 3, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1^2 + x_2^2 \leq 1\right), \\
\phi &\equiv \left(x_1 \leq 4 \wedge x_1 \geq -4 \wedge x_2 \leq 4 \wedge x_2 \geq -4\right).
\end{aligned}$$

26. Man & MacCallum (1997), Goriely [Gor01] p. 57.

$$\begin{aligned}
\dot{x_1} &= 2x_1^2x_2 - x_2, \\
\dot{x_2} &= 2x_1x_2^2 + x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1^2 + x_2^2 \leq \frac{1}{4}\right), \\
\phi &\equiv \left(x_1 \leq 3\right).
\end{aligned}$$

27. Collin (1995), Goriely [Gor01] p. 57.

$$\begin{aligned}
\dot{x_1} &= x_1^2 + 2x_2x_1 + 3x_2^2, \\
\dot{x_2} &= 2x_2\left(2x_1 + x_2\right), \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 + 2\right)^2 + \left(x_2 - 1\right)^2 \leq \frac{1}{4}\right), \\
\phi &\equiv \left(x_1 \leq 0\right).
\end{aligned}$$

28. KeYmaera problem (lics1-continuous-forward).

$$\begin{aligned}\dot{x} &= v,\\ \dot{v} &= a,\\ \dot{a} &= 0,\\ H &\equiv \text{True},\\ \psi &\equiv (v \geq 0 \wedge a \geq 0),\\ \phi &\equiv (v \geq 0).\end{aligned}$$

29. KeYmaera problem (nonlinear-diffcut).

$$\begin{aligned}\dot{x} &= (x-3)^4 + y^5,\\ \dot{y} &= y^2,\\ H &\equiv \text{True},\\ \psi &\equiv \left(x^3 \geq -1 \wedge y^5 \geq 0\right),\\ \phi &\equiv \left(x^3 \geq -1 \wedge y^5 \geq 0\right).\end{aligned}$$

30. KeYmaera problem (nonlinear1).

$$\begin{aligned}\dot{x} &= a + (x-3)^4,\\ \dot{a} &= 0,\\ H &\equiv (a \geq 0),\\ \psi &\equiv \left(x^3 \geq -1\right),\\ \phi &\equiv \left(x^3 \geq -1\right).\end{aligned}$$

31. Invariant 3−dimensional unit sphere (smooth manifold) enclosing an equilibrium at the origin.

$$\begin{aligned}\dot{x_1} &= x_1^2 - x_1\left(x_1^3 + x_2^3 + x_3^3\right),\\ \dot{x_2} &= x_2^2 - x_2\left(x_1^3 + x_2^3 + x_3^3\right),\\ \dot{x_3} &= x_3^2 - x_3\left(x_1^3 + x_2^3 + x_3^3\right),\\ H &\equiv \text{True},\\ \psi &\equiv \left(x_1 = \frac{1}{4} \wedge x_2 = \frac{1}{8} \wedge x_3 = \frac{1}{10}\right),\\ \phi &\equiv \left(x_1 \leq 10 \wedge x_2 \leq 5 \wedge x_3 > -20\right).\end{aligned}$$

32. S. Prajna PhD thesis [Pra05] (barrier certificate example) Section 2.4.1, p. 31.

$$\begin{aligned}
\dot{x_1} &= x_1, \\
\dot{x_2} &= \frac{x_1^3}{3} - x_1 - x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{3}{2} \right)^2 + x_2^2 \leq \frac{1}{4} \right), \\
\phi &\equiv \left((x_1 + 1)^2 + (x_2 + 1)^2 > \frac{4}{25} \right).
\end{aligned}$$

33. Strogatz [Str94], Example 6.3.2.

$$\begin{aligned}
\dot{x_1} &= x_1 \left(x_1^2 + x_2^2 \right) - x_1, \\
\dot{x_2} &= x_1 + x_2 \left(x_1^2 + x_2^2 \right), \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 > -\frac{4}{5} \wedge x_1 < -\frac{1}{3} \wedge x_2 < \frac{3}{2} \wedge x_2 \geq 1 \right), \\
\phi &\equiv \left(x_1 \geq -\frac{1}{3} \vee x_2 < 0 \vee 2x_2 \geq 1 \vee x_1 \leq -\frac{4}{5} \right).
\end{aligned}$$

34. R. Moussu (1982), Schlomiuk [Sch93] Example 6.3.

$$\begin{aligned}
\dot{x_1} &= x_2^3, \\
\dot{x_2} &= \frac{1}{2} x_1^2 x_2^2 - x_1^3, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 > -\frac{4}{5} \wedge x_1 < -\frac{1}{3} \wedge x_2 < \frac{1}{2} \wedge x_2 \geq 0 \right), \\
\phi &\equiv (x_1 > -2).
\end{aligned}$$

35. M. Dolov (1972), Schlomiuk [Sch93] Example 6.2.

$$\begin{aligned}
\dot{x_1} &= x_2^3 - x_2, \\
\dot{x_2} &= -x_1^2 + x_1 - (x_1 + 1) x_2^2, \\
H &\equiv \text{True}, \\
\psi &\equiv (x_1 > 0), \\
\phi &\equiv (x_1 \geq -2 \vee x_2 < -1 \vee x_2 \geq 1 \vee x_1 \leq -5).
\end{aligned}$$

36. Dumortier, Llibre & Artés [DLA06], Exercise 10.15 (ii).

$$
\begin{aligned}
\dot{x_1} &= 315x_1^7 + 477x_2x_1^6 - 113x_2^2x_1^5 + 301x_2^3x_1^4 - 300x_2^4x_1^3 - 192x_2^5x_1^2 + 128x_2^6x_1 - 16x_2^7,\\
\dot{x_2} &= x_2\left(2619x_1^6 - 99x_2x_1^5 - 3249x_2^2x_1^4 + 1085x_2^3x_1^3 + 596x_2^4x_1^2 - 416x_2^5x_1 + 64x_2^6\right),\\
H &\equiv \text{True},\\
\psi &\equiv \left(x_1 > -1 \wedge x_1 < -\frac{1}{2} \wedge x_2 \leq -\frac{1}{10} \wedge x_2 \geq -\frac{3}{10}\right),\\
\phi &\equiv \left(x_1 \leq x_2 + 1\right).
\end{aligned}
$$

37. Dumortier, Llibre & Artés [DLA06], Exercise 10.15 (i).

$$
\begin{aligned}
\dot{x_1} &= -42x_1^7 + 68x_2x_1^6 - 46x_2x_1^5 + 258x_2x_1^4 + 156x_2x_1^3 + 50x_2x_1^2 + 20x_2^6x_1 - 8x_2^7,\\
\dot{x_2} &= x_2\left(1110x_1^6 - 220x_2x_1^5 - 3182x_2x_1^4 + 478x_2^3x_1^3 + 487x_2^4x_1^2 - 102x_2^5x_1 - 12x_2^6\right),\\
H &\equiv \text{True},\\
\psi &\equiv \left(x_1 > -1 \wedge x_1 < -\frac{3}{4} \wedge x_2 \leq \frac{3}{2} \wedge x_2 \geq 1\right),\\
\phi &\equiv \left(x_1 \leq x_2 + 1\right).
\end{aligned}
$$

38. Dumortier, Llibre & Artés [DLA06], Exercise 10.11.

$$
\begin{aligned}
\dot{x_1} &= 100x_1^3 + 200x_2x_1^2 + 70x_1^2 - 100x_1 - 200x_2 - 70,\\
\dot{x_2} &= 100x_2x_1^2 + 200x_2^2x_1 + 140x_2x_1 + 146x_1 + 100x_2,\\
H &\equiv \text{True},\\
\psi &\equiv \left(x_1^2 + x_2^2 < \frac{1}{4}\right),\\
\phi &\equiv \left(x_1 > -\frac{3}{2} \wedge x_1 < \frac{3}{2}\right).
\end{aligned}
$$

39. Dumortier, Llibre & Artés [DLA06], Exercise 10.11.

$$
\begin{aligned}
\dot{x_1} &= x_1^3 + 2x_2x_1^2 + x_1^2 + x_1 + 2x_2 + 1,\\
\dot{x_2} &= x_2x_1^2 + 2x_2^2x_1 + 2x_2x_1 - x_2,\\
H &\equiv \text{True},\\
\psi &\equiv \left(\left(x_1 - 1\right)^2 + \left(x_2 + 1\right)^2 < \frac{1}{4}\right),\\
\phi &\equiv \left(x_2 < 1\right).
\end{aligned}
$$

40. Dumortier, Llibre & Artés [DLA06], Exercise 10.9.

$$\begin{aligned}
\dot{x_1} &= x_1^4 - 6x_2^2x_1^2 + 2x_2^2x_1 + \left(x_1^2 - x_2^2\right)x_1 + x_2^4,\\
\dot{x_2} &= -4x_2x_1^3 + 2x_2x_1^2 + 4x_2^3x_1 - x_2\left(x_1^2 - x_2^2\right),\\
H &\equiv \text{True},\\
\psi &\equiv \left((x_1-1)^{\,2} + (x_2+1)^{\,2} < \frac{1}{4}\right),\\
\phi &\equiv (x_2 < 1).
\end{aligned}$$

41. Dumortier, Llibre & Artés [DLA06], Exercise 10.6.

$$\begin{aligned}
\dot{x_1} &= \left(x_1^2 - 1\right)\left(x_1^2 - \left(\sqrt{5}-2\right)^2\right)\left(x_1 + \sqrt{5}x_2\right),\\
\dot{x_2} &= \left(\sqrt{5}x_1 + x_2\right)\left(x_2^2 - 1\right)\left(x_2^2 - \left(\sqrt{5}-2\right)^2\right),\\
H &\equiv \text{True},\\
\psi &\equiv \left((x_1-1)^{\,2} + x_2^2 < \frac{1}{4}\right),\\
\phi &\equiv (x_2 < 1).
\end{aligned}$$

42. Dumortier, Llibre & Artés [DLA06], Exercise 10.7.

$$\begin{aligned}
\dot{x_1} &= 2x_1\left(x_1^2 - 3\right)\left(4x_1^2 - 3\right)\left(x_1^2 + 21x_2^2 - 12\right),\\
\dot{x_2} &= x_2\left(35x_1^6 + 105x_2^2x_1^4 - 315x_1^4 - 63x_2^4x_1^2 + 378x_1^2 + 27x_2^6 - 189x_2^4 + 378x_2^2 - 216\right),\\
H &\equiv \text{True},\\
\psi &\equiv \left((x_1-1)^{\,2} + x_2^2 < \frac{1}{4}\right),\\
\phi &\equiv \left(x_1^2 + x_2^2 < 8\right).
\end{aligned}$$

43. Dumortier, Llibre & Artés [DLA06], Exercise 10.5.

$$\begin{aligned}
\dot{x_1} &= 4x_2x_1^4 - 12x_2^3x_1^2 - x_2,\\
\dot{x_2} &= -4x_1x_2^4 + 12x_1^3x_2^2 + x_1,\\
H &\equiv \text{True},\\
\psi &\equiv \left(x_1^2 + x_2^2 < \frac{1}{4}\right),\\
\phi &\equiv \left(x_1^2 + x_2^2 < 8\right).
\end{aligned}$$

44. Dumortier, Llibre & Artés [DLA06], Exercise 10.3.

$$\begin{aligned}
\dot{x_1} &= 2x_2^3 - x_1^2x_2 + x_2 + x_1^2,\\
\dot{x_2} &= -x_1^3 - x_2^2x_1 + x_1 + x_2^2,\\
H &\equiv \text{True},\\
\psi &\equiv \left(x_1 > -\frac{4}{5} \wedge x_1 < -\frac{1}{3} \wedge x_2 < 0 \wedge x_2 \geq -1\right),\\
\phi &\equiv \left(x_1^2 + x_2^2 \leq 10\right).
\end{aligned}$$

45. Dumortier, Llibre & Artés [DLA06], Exercise 5.2 (ii).

$$\begin{aligned}
\dot{x_1} &= 2x_1 - 2x_1x_2,\\
\dot{x_2} &= -x_1^2 + x_2^2 + 2x_2,\\
H &\equiv \text{True},\\
\psi &\equiv \left(x_1 > -\frac{4}{5} \wedge x_1 < -\frac{1}{3} \wedge x_2 < 0 \wedge x_2 \geq -1\right),\\
\phi &\equiv \left(x_1 + x_2 \leq 1 \wedge (x_1 \neq 0 \vee x_2 \neq 0)\right).
\end{aligned}$$

46. Dumortier, Llibre & Artés [DLA06], Exercise 5.2 (i).

$$\begin{aligned}
\dot{x_1} &= 2x_1x_2 - 4x_2 - 8,\\
\dot{x_2} &= 4x_2^2 - x_1^2,\\
H &\equiv \text{True},\\
\psi &\equiv \left(x_1 > -1 \wedge x_1 < 0 \wedge x_2 < 0 \wedge x_2 \geq -1\right),\\
\phi &\equiv \left(x_1 + x_2 \leq 1\right).
\end{aligned}$$

47. Dumortier, Llibre & Artés [DLA06], Exercise 5.1 (ii).

$$\begin{aligned}
\dot{x_1} &= x_1^2 + x_2^2 - 1,\\
\dot{x_2} &= 5\left(x_1x_2 - 1\right),\\
H &\equiv \text{True},\\
\psi &\equiv \left(x_1 > -1 \wedge x_1 < 0 \wedge x_2 < 0 \wedge x_2 \geq -1\right),\\
\phi &\equiv \left(x_1 + x_2 \leq 1\right).
\end{aligned}$$

48. Dumortier, Llibre & Artés [DLA06], Exercise 5.2.

$$\begin{aligned}
\dot{x_1} &= x_2, \\
\dot{x_2} &= x_1^5 - x_1 x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 > -1 \wedge x_1 < -\frac{1}{2} \wedge x_2 < -\frac{1}{2} \wedge x_2 \geq -1 \right), \\
\phi &\equiv (x_1 + x_2 \leq 1)\,.
\end{aligned}$$

49. Dumortier, Llibre & Artés [DLA06], Exercise 5.1.

$$\begin{aligned}
\dot{x_1} &= x_2, \\
\dot{x_2} &= -x_1^3 - x_2 x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 > -\frac{1}{2} \wedge x_1 < 0 \wedge x_2 < 0 \wedge x_2 \geq -\frac{1}{2} \right), \\
\phi &\equiv (x_2 \leq 5)\,.
\end{aligned}$$

50. Dumortier, Llibre & Artés [DLA06], Exercise 5.13.

$$\begin{aligned}
\dot{x_1} &= x_2, \\
\dot{x_2} &= 2\left(-x_2 - 1\right) x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 > -\frac{1}{2} \wedge x_1 < 0 \wedge x_2 < 0 \wedge x_2 \geq -\frac{1}{2} \right), \\
\phi &\equiv (x_2 \leq 2)\,.
\end{aligned}$$

51. Dumortier, Llibre & Artés [DLA06], Exercise 2.11.

$$\begin{aligned}
\dot{x_1} &= 6x_2^3 + 4x_1 x_2 + 3x_1^2 + 2x_1, \\
\dot{x_2} &= 2x_1^2 + 3x_2^2 x_1 + x_1 - x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 > -\frac{1}{2} \wedge x_1 < -\frac{1}{3} \wedge x_2 < 0 \wedge x_2 \geq -\frac{1}{2} \right), \\
\phi &\equiv (x_2 \leq 2)\,.
\end{aligned}$$

52. Dumortier, Llibre & Artés [DLA06], Exercise 1.11.

$$
\begin{aligned}
\dot{x_1} &= -x_1^5 - x_2^4 x_1 + 2x_1, \\
\dot{x_2} &= -x_2^3 - x_1^2 x_2 + x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 > -\frac{1}{2} \wedge x_1 < -\frac{1}{3} \wedge x_2 < 0 \wedge x_2 \geq -\frac{1}{2} \right), \\
\phi &\equiv \left(x_1 + x_2 \leq 0 \right).
\end{aligned}
$$

53. Dumortier, Llibre & Artés [DLA06], Exercise 1.11.

$$
\begin{aligned}
\dot{x_1} &= -x_1^5 - x_2^4 x_1 + 2x_1, \\
\dot{x_2} &= -x_2^3 - x_1^2 x_2 + x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 > -\frac{1}{2} \wedge x_1 < -\frac{1}{3} \wedge x_2 < 0 \wedge x_2 \geq -\frac{1}{2} \right), \\
\phi &\equiv \left(x_1^2 + x_2^2 \leq 5 \right).
\end{aligned}
$$

54. Dumortier, Llibre & Artés [DLA06], Exercise 1.9.

$$
\begin{aligned}
\dot{x_1} &= x_1 \left(-x_1^2 - x_2^2 + 1 \right) + x_2 \left(\left(x_1^2 - 1 \right)^2 + x_2^2 \right), \\
\dot{x_2} &= x_2 \left(-x_1^2 - x_2^2 + 1 \right) - x_2 \left(\left(x_1^2 - 1 \right)^2 + x_2^2 \right), \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 > -\frac{1}{2} \wedge x_1 < -\frac{1}{3} \wedge x_2 < 0 \wedge x_2 \geq -\frac{1}{2} \right), \\
\phi &\equiv \left(x_1 < 0 \right).
\end{aligned}
$$

55. Dumortier, Llibre & Artés [DLA06], Exercise 1.9.

$$
\begin{aligned}
\dot{x_1} &= x_1 \left(-x_1^2 - x_2^2 + 1 \right) + x_2 \left(\left(x_1^2 - 1 \right)^2 + x_2^2 \right), \\
\dot{x_2} &= x_2 \left(-x_1^2 - x_2^2 + 1 \right) - x_2 \left(\left(x_1^2 - 1 \right)^2 + x_2^2 \right), \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 > -\frac{1}{2} \wedge x_1 < -\frac{1}{3} \wedge x_2 < 0 \wedge x_2 \geq -\frac{1}{2} \right), \\
\phi &\equiv \left(x_1^2 + 2x_2^2 \leq 2 \right).
\end{aligned}
$$

56. Dumortier, Llibre & Artés [DLA06], Example 1.38 (ii).

$$\begin{aligned}
\dot{x_1} &= -x_1^3 - x_2^3, \\
\dot{x_2} &= x_1^3 - x_2^3, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 > -\frac{1}{2} \wedge x_1 < -\frac{1}{3} \wedge x_2 < 0 \wedge x_2 \geq -\frac{1}{2} \right), \\
\phi &\equiv (x_1 \leq x_2 + 1)\,.
\end{aligned}$$

57. Wiggins [Wig03] Exercise 18.7.3 (d).

$$\begin{aligned}
\dot{x_1} &= 2x_1 + 2x_2, \\
\dot{x_2} &= x_1^4 + x_1 + x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 > -\frac{1}{2} \wedge x_1 < -\frac{1}{3} \wedge x_2 < 0 \wedge x_2 \geq -\frac{1}{2} \right), \\
\phi &\equiv (x_2 > -2)\,.
\end{aligned}$$

58. Wiggins [Wig03] Exercise 18.7.3 (g).

$$\begin{aligned}
\dot{x_1} &= -x_2 x_1 - x_1 - x_2, \\
\dot{x_2} &= 2x_2 x_1 + 2x_1 + x_2, \\
H &\equiv (x_1 > -2 \wedge x_1 < 2 \wedge x_2 > -2 \wedge x_2 < 2)\,, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3} \right)^2 + \left(x_2 - \frac{1}{3} \right)^2 < \frac{1}{4} \right), \\
\phi &\equiv (x_2 > -1)\,.
\end{aligned}$$

59. Wiggins [Wig03] Exercise 18.7.3 (f).

$$\begin{aligned}
\dot{x_1} &= x_2^3 + 3x_2 - 2x_1, \\
\dot{x_2} &= x_1^3 + 2x_1 - 3x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3} \right)^2 + \left(x_2 - \frac{1}{3} \right)^2 < \frac{1}{16} \right), \\
\phi &\equiv (x_2 > -1)\,.
\end{aligned}$$

60. Wiggins [Wig03] Exercise 18.7.3 (n).

$$\begin{aligned}
\dot{x_1} &= x_1^4 x_2^5 + x_1^2 x_2 + 2x_2 - x_1, \\
\dot{x_2} &= x_1^8 x_2^9 - x_1^4 x_2^6 - x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3} \right)^2 + \left(x_2 - \frac{1}{3} \right)^2 < \frac{1}{16} \right), \\
\phi &\equiv \left(x_2 > -1 \wedge x_1 > -1 \right).
\end{aligned}$$

61. Wiggins [Wig03] Exercise 18.1.2.

$$\begin{aligned}
\dot{x_1} &= -x_1^6 - x_2 x_1, \\
\dot{x_2} &= x_1^2 - x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3} \right)^2 + \left(x_2 - \frac{1}{3} \right)^2 < \frac{1}{16} \right), \\
\phi &\equiv \left(x_2 > -1 \right).
\end{aligned}$$

62. Thieme (1994), Wiggins [Wig03] Exercise 17.1.2.

$$\begin{aligned}
\dot{x_1} &= \left(x_1 + 2 \right) \left(x_2 - \left(1 - x_1 \right) x_1 \right), \\
\dot{x_2} &= -x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3} \right)^2 + \left(x_2 - \frac{1}{3} \right)^2 < \frac{1}{16} \right), \\
\phi &\equiv \left(x_1 > -\frac{5}{2} \right).
\end{aligned}$$

63. Strogatz [Str94], Example 7.3.5.

$$\begin{aligned}
\dot{x_1} &= \left(x_1^2 + 2x_2^2 \right) x_1 - x_1 - x_2, \\
\dot{x_2} &= x_1 - x_2 + x_2 \left(x_1^2 + 2x_2^2 \right), \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3} \right)^2 + \left(x_2 - \frac{1}{3} \right)^2 < \frac{1}{16} \right), \\
\phi &\equiv \left(x_1 > -2 \wedge x_2 > -1 \wedge \left(x_1 \neq 0 \vee x_2 \neq 0 \right) \right).
\end{aligned}$$

64. Strogatz [Str94], Example 6.6.1.

$$\begin{aligned}
\dot{x_1} &= \left(1 - x_1^2\right) x_2, \\
\dot{x_2} &= 1 - x_2^2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3}\right)^2 + \left(x_2 - \frac{1}{3}\right)^2 < \frac{1}{16}\right), \\
\phi &\equiv \left(x_1 > -2 \wedge x_1 < 2\right).
\end{aligned}$$

65. Strogatz [Str94], Example 6.3.9.

$$\begin{aligned}
\dot{x_1} &= x_2^3 - 4x_1, \\
\dot{x_2} &= x_2^3 - x_2 - 3x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3}\right)^2 + \left(x_2 - \frac{1}{3}\right)^2 < \frac{1}{16}\right), \\
\phi &\equiv \left(x_1 > -2 \wedge x_1 < 2\right).
\end{aligned}$$

66. Strogatz [Str94], Example 6.3.6.

$$\begin{aligned}
\dot{x_1} &= x_1 x_2 - 1, \\
\dot{x_2} &= x_1 - x_2^3, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3}\right)^2 + \left(x_2 - \frac{1}{3}\right)^2 < \frac{1}{16}\right), \\
\phi &\equiv \left(x_1 > -2 \wedge x_1 < 2\right).
\end{aligned}$$

67. Strogatz [Str94], Example 6.3.4.

$$\begin{aligned}
\dot{x_1} &= -x_1^3 + x_1 + x_2, \\
\dot{x_2} &= -x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3}\right)^2 + \left(x_2 - \frac{1}{3}\right)^2 < \frac{1}{16}\right), \\
\phi &\equiv \left(x_1 > -2 \wedge x_1 < 2\right).
\end{aligned}$$

68. Strogatz [Str94], Example 6.2.2.

$$\begin{aligned}
\dot{x_1} &= x_2, \\
\dot{x_2} &= x_2\left(-x_1^2 - x_2^2 + 1\right) - x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3}\right)^2 + \left(x_2 - \frac{1}{3}\right)^2 < \frac{1}{16}\right), \\
\phi &\equiv \left(x_1 > -2 \wedge x_1 < 2 \wedge x_1^2 + x_2^2 \neq 0\right).
\end{aligned}$$

69. Strogatz [Str94], Example 6.1.5.

$$\begin{aligned}
\dot{x_1} &= x_1\left(-x_1 - x_2 + 2\right), \\
\dot{x_2} &= x_1 - x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3}\right)^2 + 2\left(x_2 - \frac{1}{3}\right)^2 < \frac{1}{25}\right), \\
\phi &\equiv \left(x_1 > -2 \wedge x_1 < 2\right).
\end{aligned}$$

70. Strogatz [Str94], Example 6.1.3.

$$\begin{aligned}
\dot{x_1} &= x_1\left(x_1 - x_2\right), \\
\dot{x_2} &= \left(2x_1 - x_2\right)x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3}\right)^2 + 2\left(x_2 - \frac{1}{3}\right)^2 < \frac{1}{25}\right), \\
\phi &\equiv \left(x_1 < 2\right).
\end{aligned}$$

71. Strogatz [Str94], Example 6.1.6.

$$\begin{aligned}
\dot{x_1} &= x_1^2 - x_2, \\
\dot{x_2} &= x_1 - x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{1}{3}\right)^2 + 2\left(x_2 - \frac{1}{2}\right)^2 < \frac{1}{25}\right), \\
\phi &\equiv \left(x_1 > -2 \wedge x_1 < 2\right).
\end{aligned}$$

72. Strogatz [Str94], Example 6.1.9.

$$\begin{aligned}
\dot{x_1} &= 2x_1x_2,\\
\dot{x_2} &= x_2^2 - x_1^2,\\
H &\equiv \text{True},\\
\psi &\equiv \left(\left(x_1 - \frac{1}{3}\right)^2 + x_2^2 < \frac{1}{25}\right),\\
\phi &\equiv (x_1 > -2)\,.
\end{aligned}$$

73. Strogatz [Str94], Example 6.1.9.

$$\begin{aligned}
\dot{x_1} &= 2x_1x_2,\\
\dot{x_2} &= x_2^2 - x_1^2,\\
H &\equiv \text{True},\\
\psi &\equiv \left(\left(x_1 - \frac{1}{3}\right)^2 + 2x_2^2 < \frac{1}{25}\right),\\
\phi &\equiv (x_1 > -2 \wedge x_1 \leq 10 \wedge (x_1 \neq 0 \vee x_2 \neq 0))\,.
\end{aligned}$$

74. Borrelli & Coleman (1987), Strogatz [Str94], Example 6.1.10.

$$\begin{aligned}
\dot{x_1} &= x_2^2 + x_2,\\
\dot{x_2} &= \frac{6x_2^2}{5} - x_1x_2 + \frac{x_2}{5} - \frac{x_1}{2},\\
H &\equiv \text{True},\\
\psi &\equiv \left(4\left(x_1 - \frac{1}{3}\right)^2 + x_2^2 < \frac{1}{16}\right),\\
\phi &\equiv (x_1 > -2 \wedge (x_1 \neq 0 \vee x_2 \neq 0))\,.
\end{aligned}$$

75. Strogatz [Str94], Example 6.8.3.

$$\begin{aligned}
\dot{x_1} &= x_1^2x_2,\\
\dot{x_2} &= x_1^2 - x_2^2,\\
H &\equiv \text{True},\\
\psi &\equiv \left(2\left(x_1 - \frac{1}{3}\right)^2 + x_2^2 < \frac{1}{16}\right),\\
\phi &\equiv (x_1 > -2)\,.
\end{aligned}$$

76. Strogatz [Str94], Example 6.6.1.

$$\begin{aligned}
\dot{x_1} &= x_2 - x_2^3, \\
\dot{x_2} &= x_1 - x_2^2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 - \frac{3}{4} \right)^2 + x_2^2 < \frac{1}{16} \right), \\
\phi &\equiv (x_1 > -2)\,.
\end{aligned}$$

77. Stable limit cycle on the unit circle 1.

$$\begin{aligned}
\dot{x_1} &= -x_1^3 - x_2^2 x_1 + x_1 + x_2, \\
\dot{x_2} &= -x_2^3 - x_1^2 x_2 + x_2 - x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1^2 + x_2^2 < \frac{1}{16} \right), \\
\phi &\equiv (x_1 > -2 \wedge x_2 \leq 2)\,.
\end{aligned}$$

78. Stable limit cycle on the unit circle 2.

$$\begin{aligned}
\dot{x_1} &= -x_1^3 - x_2^2 x_1 + x_1 + x_2, \\
\dot{x_2} &= -x_2^3 - x_1^2 x_2 + x_2 - x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1^2 + x_2^2 < \frac{1}{16} \right), \\
\phi &\equiv (x_1 \geq -1 \wedge x_2 \leq 1 \wedge x_2 \geq -1 \wedge x_1 \leq 1)\,.
\end{aligned}$$

79. Tiwari & Khanna [TK04], Example 8, Pendulum.

$$\begin{aligned}
\dot{x_1} &= x_2, \\
\dot{x_2} &= \frac{x_1^3}{6} - x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(2 \left(x_1 - \frac{1}{2} \right)^2 + x_2^2 < \frac{1}{8} \right), \\
\phi &\equiv (x_1 > -2)\,.
\end{aligned}$$

80. M. Jirstrand PhD. thesis [Jir98], Example 12.1, page 215.

$$\begin{aligned}
\dot{x_1} &= -x_1 + \frac{5x_2}{2} + \frac{9x_3}{10}, \\
\dot{x_2} &= -\frac{3x_1}{2} - x_2 - \frac{x_3}{2}, \\
\dot{x_3} &= -\frac{1}{5}\left(4x_3\right), \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 = 1 \wedge x_2^2 < 1 \wedge x_3 < 1 \wedge x_3 > 0\right), \\
\phi &\equiv \left(-\frac{1}{2} < x_3 \wedge x_3 < \frac{5}{2}\right).
\end{aligned}$$

81. Bhatia & Szegő [BS70], page 68, Example 2.4.

$$\begin{aligned}
\dot{x_1} &= 2x_1^3x_2^2 - x_1, \\
\dot{x_2} &= -x_2, \\
H &\equiv \left(x_1^2x_2^2 < 1\right), \\
\psi &\equiv \left(x_1^2 + \left(x_2 - \frac{1}{2}\right)^{\,2} < \frac{1}{24}\right), \\
\phi &\equiv \left(x_1 > -2 \wedge x_2 > -1\right).
\end{aligned}$$

82. K. Forsman PhD. thesis [For91], page 99, Example 6.1.

$$\begin{aligned}
\dot{x_1} &= 2x_1^2x_2 - x_1, \\
\dot{x_2} &= -x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1^2 + \left(x_2 - 2\right)^{\,2} < \frac{1}{24}\right), \\
\phi &\equiv \left(x_1 > -2 \wedge x_2 > -1\right).
\end{aligned}$$

83. K. Forsman PhD. thesis [For91], page 101, Example 6.3.

$$\begin{aligned}
\dot{x_1} &= 2x_2^2 - x_1, \\
\dot{x_2} &= x_1^2 + x_3^2 - x_2, \\
\dot{x_3} &= -x_1^2 - x_3, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1^2 + 4x_2^2 + x_3^2 < \frac{1}{100}\right), \\
\phi &\equiv \left(x_1 < 5 \wedge x_2 \leq 2 \wedge x_3^2 \leq 25\right).
\end{aligned}$$

84. K. Forsman PhD. thesis [For91], page 119, Example 6.13.

$$\begin{aligned}
\dot{x_1} &= -x_2 x_1 - x_1, \\
\dot{x_2} &= -x_1^3 - x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1^2 + (x_2 - 1)^2 < \frac{1}{8} \right), \\
\phi &\equiv (x_2 > -1).
\end{aligned}$$

85. K. Forsman PhD. thesis [For91], page 119, Example 6.14.

$$\begin{aligned}
\dot{x_1} &= x_2^4 - 2x_1, \\
\dot{x_2} &= 3x_1 x_2^3 - x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1^2 + (x_2 - 1)^2 < \frac{1}{8} \right), \\
\phi &\equiv (x_2 > -1).
\end{aligned}$$

86. Alongi & Nelson [AN07], page 143, Example 4.1.9.

$$\begin{aligned}
\dot{x_1} &= x_1 x_3, \\
\dot{x_2} &= x_2 x_3, \\
\dot{x_3} &= -x_1^2 - x_2^2, \\
H &\equiv \left(x_1^2 + x_2^2 + x_3^2 = 1 \right), \\
\psi &\equiv (x_1 = 1 \wedge x_2 = 0 \wedge x_3 = 0), \\
\phi &\equiv (x_3 \leq 0 \wedge x_1 \leq 1).
\end{aligned}$$

87. C. Wang, S. Lall & M. West [WLW13], Example 2.

$$\begin{aligned}
\dot{x_1} &= x_2, \\
\dot{x_2} &= -\frac{x_1}{5} - \frac{x_2}{10}, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 + \frac{\pi}{2} \right)^2 + x_2^2 \leq \frac{1}{10} \right), \\
\phi &\equiv (x_2 > -1).
\end{aligned}$$

88. Unreachable equilibrium.

$$\begin{aligned}
\dot{x_1} &= -x_1^3 - x_2^2 x_1 + x_1 + x_2, \\
\dot{x_2} &= -x_2^3 - x_1^2 x_2 + x_2 - x_1, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(x_1 = 1 \wedge x_2 = \frac{1}{4} \right), \\
\phi &\equiv \left(x_1 \neq 0 \vee x_2 \neq 0 \right).
\end{aligned}$$

89. A.A. Ahmadi, M. Kristić & P. A. Parrilo [AKP11], Theorem 1.

$$\begin{aligned}
\dot{x_1} &= x_1 x_2 - x_1, \\
\dot{x_2} &= -x_2, \\
H &\equiv \text{True}, \\
\psi &\equiv \left(\left(x_1 + \frac{7}{10} \right)^2 + \left(x_2 - \sqrt{7} \right)^2 \leq \frac{1}{32} \right), \\
\phi &\equiv \left(x_1 \geq -\frac{5}{2} \right).
\end{aligned}$$

90. S. Ratschan & Z. She [RS07], p. 14, Example FOCUS.

$$\begin{aligned}
\dot{x_1} &= x_1 - x_2, \\
\dot{x_2} &= x_1 + x_2, \\
H &\equiv \left(0 \leq x_1 \wedge x_1 \leq 4 \wedge 0 \leq x_2 \wedge x_2 \leq 4 \right), \\
\psi &\equiv \left(\frac{5}{2} \leq x_1 \wedge x_1 \leq 3 \wedge x_2 = 0 \right), \\
\phi &\equiv \left(x_1 > 2 \right).
\end{aligned}$$

91. S. Ratschan & Z. She [RS07], p. 14, Example 1-FLOW.

$$
\begin{aligned}
&\dot{x_1} = 1,\\
&\dot{x_2} = 1,\\
&\dot{x_3} = 1,\\
&H \equiv (0 \le x_1 \wedge x_1 \le 2 \wedge 0 \le x_2 \wedge x_2 \le 2 \wedge 0 \le x_3 \wedge x_3 \le 4)\,,\\
&\psi \equiv (0 \le x_1 \wedge x_1 \le 1 \wedge x_2 = 0 \wedge x_3 = 0)\,,\\
&\phi \equiv \Big((x_2 > 2 \wedge 0 \le x_1 \wedge x_1 \le 2)\\
&\quad \vee (0 \le x_1 \wedge x_1 \le 2 \wedge x_2 \le 1)\\
&\quad \vee (x_3 \ge 1 \wedge 1 < x_2 \wedge 0 \le x_1 \wedge x_1 \le 2 \wedge x_2 \le 2)\\
&\quad \vee (1 < x_2 \wedge x_3 < 0 \wedge 0 \le x_1 \wedge x_1 \le 2 \wedge x_2 \le 2)\\
&\quad \vee x_1 > 2 \vee x_1 < 0\Big).
\end{aligned}
$$

92. S. Ratschan & Z. She [RS07], p. 14, Example 1-CLOCK.

$$
\begin{aligned}
&\dot{x_1} = x_2^2 - \frac{11x_2}{2},\\
&\dot{x_2} = 6x_1 - x_1^2,\\
&\dot{x_3} = 1,\\
&H \equiv (1 \le x_1 \wedge x_1 \le 5 \wedge 1 \le x_2 \wedge x_2 \le 5 \wedge 0 \le x_3 \wedge x_3 \le 4)\,,\\
&\psi \equiv \left(4 \le x_1 \wedge x_1 \le \frac{9}{2} \wedge x_2 = 1 \wedge x_3 = 0\right),\\
&\phi \equiv \Big((x_2 \ge 3 \wedge x_1 < 2 \wedge 1 \le x_1)\\
&\quad \vee (x_1 < 2 \wedge 1 \le x_1 \wedge x_2 \le 2)\\
&\quad \vee (x_3 > 4 \wedge 2 < x_2 \wedge x_1 < 2 \wedge x_2 < 3 \wedge 1 \le x_1)\\
&\quad \vee (2 < x_2 \wedge x_1 < 2 \wedge x_2 < 3 \wedge x_3 < 2 \wedge 1 \le x_1)\\
&\quad \vee x_1 \ge 2 \vee x_1 < 1\Big).
\end{aligned}
$$

93. S. Gulwani & A. Tiwari [GT08], Example 8, phytoplankton growth model.

$$\begin{aligned}
\dot{x_1} &= -\frac{1}{4}x_2x_1 - x_1 + 1,\\
\dot{x_2} &= x_2\left(2x_3 - 1\right),\\
\dot{x_3} &= \frac{x_1}{4} - 2x_3^2,\\
H &\equiv \text{True},\\
\psi &\equiv \left(0 \leq x_1 \wedge x_1 \leq 1 \wedge x_2 = \frac{1}{2} \wedge x_3 = \frac{1}{3}\right),\\
\phi &\equiv \left(x_1 \leq 3 \wedge x_2 \leq 2 \wedge x_3 \geq -1\right).
\end{aligned}$$

94. S. Gulwani & A. Tiwari [GT08], Example 1, adaptive cruise controller.

$$\begin{aligned}
\dot{v} &= a,\\
\dot{vf} &= af,\\
\dot{a} &= -3a + gap - 3(v - vf) - v - 10,\\
\dot{gap} &= vf - v,\\
\dot{af} &= 0,\\
H &\equiv (v \geq 0 \wedge vf \geq 0 \wedge -2 \leq a \wedge a \leq 5 \wedge -2 \leq af \wedge af \leq 5),\\
\psi &\equiv (gap = 5 \wedge v = vf \wedge a = 0),\\
\phi &\equiv (gap > 0).
\end{aligned}$$

95. S. Ratschan & Z. She [RS06], Example 11.

$$\begin{aligned}
\dot{x_1} &= \frac{4x_2}{5} + \frac{3x_3}{5} - \frac{9}{5},\\
\dot{x_2} &= \frac{4x_1}{5} + \frac{7x_3}{10} - \frac{76}{5},\\
\dot{x_3} &= \frac{3x_1}{5} + \frac{7x_2}{10} - \frac{9}{5},\\
H &\equiv \left(15 \leq x_1 \wedge x_1 \leq 24 \wedge 15 \leq x_2 \wedge x_2 \leq 24 \wedge 15 \leq x_3 \wedge x_3 \leq 24\right),\\
\psi &\equiv \left(19 \leq x_1 \wedge x_1 \leq 20 \wedge 19 \leq x_2 \wedge x_2 \leq 20 \wedge 19 \leq x_3 \wedge x_3 \leq 20\right),\\
\phi &\equiv \left(x_1 > 21 \vee x_2 > 20 \vee x_3 < \frac{45}{2}\right).
\end{aligned}$$

96. Lorenz system ($\sigma = 10, \rho = 28, \beta = \frac{8}{9}$).

$$\begin{aligned}
\dot{x_1} &= 10\,(x_2 - x_1)\,,\\
\dot{x_2} &= x_1\,(28 - x_3) - x_2,\\
\dot{x_3} &= x_1 x_2 - \frac{8x_3}{9},\\
H &\equiv \text{True},\\
\psi &\equiv \left(x_1 = 0 \wedge x_2 = 1 \wedge x_3 = \frac{1}{10}\right),\\
\phi &\equiv (x_3 \geq -20)\,.
\end{aligned}$$

97. Shimizu-Morioka system ($a = 1, b = 1$).

$$\begin{aligned}
\dot{x_1} &= x_2,\\
\dot{x_2} &= -x_3 x_1 + x_1 - x_2,\\
\dot{x_3} &= x_1^2 - x_3,\\
H &\equiv \text{True},\\
\psi &\equiv (x_1 = 5 \wedge x_2 = 3 \wedge x_3 = -4)\,,\\
\phi &\equiv (x_3 \geq -5)\,.
\end{aligned}$$

98. Michelson system ($c = 2$).

$$\begin{aligned}
\dot{x_1} &= x_2,\\
\dot{x_2} &= x_3,\\
\dot{x_3} &= -\frac{x_1^2}{2} - x_2 + 4,\\
H &\equiv (-20 \leq x_1 \wedge x_1 \leq 20 \wedge -20 \leq x_2 \wedge x_2 \leq 20 \wedge -20 \leq x_3 \wedge x_3 \leq 20)\,,\\
\psi &\equiv (x_1 = -2 \wedge x_2 = 0 \wedge x_3 = -1)\,,\\
\phi &\equiv (x_3 \leq 10)\,.
\end{aligned}$$

99. Goryachev–Chaplygin Top, Goriely [Gor01] p. 63.

$$
\begin{aligned}
\dot{x_1} &= \frac{3x_2x_3}{4},\\
\dot{x_2} &= \frac{1}{8}\left(3x_6 - 6x_1x_3\right),\\
\dot{x_3} &= -\frac{1}{2}\left(3x_5\right),\\
\dot{x_4} &= x_3x_5 - x_2x_6,\\
\dot{x_5} &= x_1x_6 - x_3x_4,\\
\dot{x_6} &= x_2x_4 - x_1x_5,\\
H &\equiv \left(x_4^2 + x_5^2 + x_6^2 = 1\right),\\
\psi &\equiv \left(x_1 = -1 \wedge x_2 = 0 \wedge x_3 = -1 \wedge x_4 = \frac{1}{2} \wedge x_5 = -\frac{1}{2} \wedge x_6 = \frac{1}{\sqrt{2}}\right),\\
\phi &\equiv \left(4x_1^2 + 3x_2^2 + 2x_3^2 < 13 - 6x_4\right).
\end{aligned}
$$

100. Klapper–Rado–Tabor model, Goriely [Gor01] p. 48.

$$
\begin{aligned}
\dot{x_1} &= x_4,\\
\dot{x_2} &= x_5,\\
\dot{x_3} &= x_6,\\
\dot{x_4} &= \frac{x_1x_3}{3} - \frac{2x_1}{3},\\
\dot{x_5} &= \frac{x_1}{3} + \frac{2x_2}{3} + \frac{x_2x_3}{3},\\
\dot{x_6} &= x_3^2 + 8x_5 - 4,\\
H &\equiv \text{True},\\
\psi &\equiv \left(x_2 = 2 \wedge x_3 = 0 \wedge x_4 = \frac{1}{2} \wedge x_5 = 1 \wedge x_6 = -2\right),\\
\phi &\equiv \left(2x_3^3 + 72x_5x_3 - 24x_3 - 72x_4 - 144x_5 < 3x_6^2 + 24x_2x_6 + 288\right).
\end{aligned}
$$

Bibliography

[ABW97] Jitendra Agarwal, David I. Blockley, and Norman J. Woodman. Qualitative analysis of non-linear dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 145(1–2):135 – 145, 1997.

[ACHH92] Rajeev Alur, Costas Courcoubetis, Thomas A. Henzinger, and Pei-Hsin Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. pages 209–229. Springer-Verlag, 1992.

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2):183–235, 1994.

[ADG07] Eugene Asarin, Thao Dang, and Antoine Girard. Hybridization methods for the analysis of nonlinear systems. *Acta Inf.*, 43(7):451–476, 2007.

[ADI03] Rajeev Alur, Thao Dang, and Franjo Ivančić. Progress on reachability analysis of hybrid systems using predicate abstraction. In *Hybrid Systems: Computation and Control, 6th International Workshop, HSCC 2003 Prague, Czech Republic, April 3-5, 2003, Proceedings*, pages 4–19, 2003.

[ADI06] Rajeev Alur, Thao Dang, and Franjo Ivančić. Predicate abstraction for reachability analysis of hybrid systems. *ACM Trans. Embedded Comput. Syst.*, 5(1):152–199, 2006.

[AHLP00] Rajeev Alur, Thomas A Henzinger, Gerardo Lafferriere, and George J Pappas. Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, 88(7):971–984, 2000.

[AHS01] Erika Ábrahám-Mumm, Ulrich Hannemann, and Martin Steffen. Verification of hybrid systems: formalization and proof rules in PVS. In *Proceedings of the Seventh IEEE International Conference on Engineering of Complex Computer Systems*, pages 48–57, 2001.

[AKP11] Amir Ali Ahmadi, Miroslav Krstić, and Pablo A. Parrilo. A globally asymptotically stable polynomial vector field with no polynomial Lyapunov function. In *CDC-ECE*, pages 7579–7580, 2011.

[Alu99] Rajeev Alur. Timed automata. In *Computer Aided Verification, 11th International Conference, CAV '99, Trento, Italy, July 6-10, 1999, Proceedings*, pages 8–22, 1999.

[AN07] John M. Alongi and Gail Susan Nelson. *Recurrence and Topology*, volume 85 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, Rhode Island, 2007.

[AP92] D.K. Arrowsmith and C.M. Place. *Dynamical Systems. Differential equations, maps and chaotic behaviour.* Chapman & Hall, 1992.

[AP10] Behzad Akbarpour and Lawrence C. Paulson. MetiTarski: An automatic theorem prover for real-valued special functions. *Journal of Automated Reasoning*, 44(3):175–205, 2010.

[AS85] Bowen Alpern and Fred B. Schneider. Defining liveness. *Information processing letters*, 21(4):181–185, 1985.

[AS87] Bowen Alpern and Fred B. Schneider. Recognizing safety and liveness. *Distributed Computing*, 2(3):117–126, 1987.

[BBJ15] Stanley Bak, Sergiy Bogomolov, and Taylor T. Johnson. HYST: a source transformation and translation tool for hybrid automaton models. In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control, HSCC'15, Seattle, WA, USA, April 14-16, 2015*, pages 128–133, 2015.

[BCC+03] Armin Biere, Alessandro Cimatti, Edmund M. Clarke, Ofer Strichman, and Yunshan Zhu. Bounded model checking. *Advances in Computers*, 58:117–148, 2003.

[BCM+90] J. R. Burch, E.M. Clarke, K. L. McMillan, D.L. Dill, and L. J. Hwang. Symbolic model checking: 10^{20} states and beyond. In *Logic in Computer Science, 1990. LICS '90, Proceedings., Fifth Annual IEEE Symposium on Logic in Computer Science*, pages 428–439, 1990.

[Bla99] Franco Blanchini. Set invariance in control. *Automatica*, 35(11):1747–1767, 1999.

[BM98] Martin Berz and Kyoko Makino. Verified integration of ODEs and flows using differential algebraic methods on high-order Taylor models. *Reliable Computing*, 4(4):361–369, 1998.

[BM08] Franco Blanchini and Stefano Miani. *Set-theoretic Methods in Control*. Systems & Control : Foundations & Applications. Birkhäuser Boston, 2008.

[Bon69] Jean-Michel Bony. Principe du maximum, inégalité de Harnack et unicité du problème de Cauchy pour les opérateurs elliptiques dégénérés. *Annales de l'institut Fourier*, 19(1):277–304, 1969.

[BPR96] Saugata Basu, Richard Pollack, and Marie-Françoise Roy. On the combinatorial and algebraic complexity of quantifier elimination. *Journal of the ACM (JACM)*, 43(6):1002–1045, 1996.

[BPR06] S. Basu, R. Pollack, and M.F. Roy. *Algorithms in Real Algebraic Geometry*. Algorithms and Computation in Mathematics. Springer, second edition, 2006.

[Bre70] Haim Brezis. On a characterization of flow-invariant sets. *Communications on Pure and Applied Mathematics*, 23(2):261–263, 1970.

[Bro05] Christopher W Brown. On quantifer elimination by virtual term substitution. Technical Report USNA-CS-TR-2005-07, U.S. Naval Academy, Computer Science Department, 572M Holloway Rd Stop 9F, Annapolis, MD 21403, August 2005.

[BS70] Nam Parshad Bhatia and George Philip Szegő. *Stability Theory of Dynamical Systems*, volume 161 of *Die Grundlehren der mathematischen Wissenschaften in Einzeldarstellungen mit besonderer Berücksichtigung der Anwendungsgebiete*. Springer-Verlag, 1970.

[Buc06] Bruno Buchberger. Bruno Buchberger's PhD thesis 1965: An algorithm for finding the basis elements of the residue class ring of a zero dimensional polynomial ideal. *Journal of Symbolic Computation*, 41(3):475–511, 2006.

[Car13] Rebekah Anne Carter. *Verification of liveness properties on hybrid dynamical systems*. PhD thesis, University of Manchester, School of Computer Science, 2013.

[CÁS12] Xin Chen, Erika Ábrahám, and Sriram Sankaranarayanan. Taylor model flowpipe construction for non-linear hybrid systems. In *Proceedings of the 33rd IEEE Real-Time Systems Symposium, RTSS 2012, San Juan, PR, USA, December 4-7, 2012*, pages 183–192, 2012.

[CÁS13] Xin Chen, Erika Ábrahám, and Sriram Sankaranarayanan. Flow*: An analyzer for non-linear hybrid systems. In *Computer Aided Verification - 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings*, pages 258–263, 2013.

[CE82] Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In *Logic of Programs, Workshop*, pages 52–71, London, UK, 1982. Springer-Verlag.

[CES86] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Trans. Program. Lang. Syst.*, 8(2):244–263, April 1986.

[CFH+03] Edmund M. Clarke, Ansgar Fehnker, Zhi Han, Bruce H. Krogh, Joël Ouaknine, Olaf Stursberg, and Michael Theobald. Abstraction and counterexample-guided refinement in model checking of hybrid systems. *International Journal of Foundations of Computer Science*, 14(4):583–604, 2003.

[CH91] George E. Collins and H. Hong. Partial cylindrical algebraic decomposition for quantifier elimination. *J. Symb. Comput.*, 12(3):299–328, 1991.

[Chi06] Carmen Chicone. *Ordinary Differential Equations with Applications*, volume 34 of *Texts in Applied Mathematics*. Springer, second edition, 2006.

[CJ98] B.F. Caviness and J.R. Johnson. *Quantifier Elimination and Cylindrical Algebraic Decomposition*. Texts and monographs in symbolic computation. Springer, 1998.

[CLO10] David Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Undergraduate Texts in Mathematics. Springer, third edition, 2010.

[CLPW09] Colin Christopher, Jaume Llibre, Chara Pantazi, and Sebastian Walcher. Inverse problems for invariant algebraic curves: explicit computations. *Proceedings of the Royal Society of Edinburgh: Section A Mathematics*, 139(2):287, 2009.

[CNL12] Rebekah Carter and Eva M. Navarro-López. Dynamically-driven timed automaton abstractions for proving liveness of continuous systems. In Marcin Jurdziński and Dejan Ničković, editors, *Formal Modeling and Analysis of Timed Systems*, volume 7595 of *Lecture Notes in Computer Science*, pages 59–74. Springer Berlin Heidelberg, 2012.

[CNV07] Ovidiu Cârjă, Mihai Necula, and Ioan I. Vrabie. *Viability, Invariance and Applications*. Number 207 in North-Holland Mathematics Studies. Elsevier Science, 2007.

[Col75] George E. Collins. Hauptvortrag: Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In *Automata Theory and Formal Languages, 2nd GI Conference, Kaiserslautern*, pages 134–183, May 1975.

[Com] Computer Assisted Proofs In Dynamics Group, Division of Computational Mathematics, Jagiellonian University, Kraków, Poland . CAPD library. Online `http://capd.ii.uj.edu.pl/`.

[Cra72] Michael G Crandall. A generalization of Peano's existence theorem and flow invariance. *Proceedings of the American Mathematical Society*, 36(1):151–155, 1972.

[CSÁ14] Xin Chen, Sriram Sankaranarayanan, and Erika Ábrahám. Under-approximate flowpipes for non-linear continuous systems. In *Formal Methods in Computer-Aided Design, FMCAD 2014, Lausanne, Switzerland, October 21-24, 2014*, pages 59–66, 2014.

[Dar78] Jean-Gaston Darboux. Mémoire sur les équations différentielles algébriques du premier ordre et du premier degré. *Bulletin des Sciences Mathématiques et Astronomiques*, 2(1):151–200, 1878.

[dBBCK08] Mario di Bernardo, Christopher J. Budd, Alan R. Champneys, and Piotr Kowalczyk. *Piecewise-smooth Dynamical Systems: Theory and Applications*. Springer, 2008.

[Dem70] Vladimir F. Demyanov. The solution of minimaximin problems. *USSR Computational Mathematics and Mathematical Physics*, 10(3):44 – 55, 1970.

[DGM11] Thao Dang, Colas Le Guernic, and Oded Maler. Computing reachable states for nonlinear biological models. *Theor. Comput. Sci.*, 412(21):2095–2107, 2011.

[DH88] James H. Davenport and Joos Heintz. Real quantifier elimination is doubly exponential. *J. Symb. Comput.*, 5(1/2):29–35, 1988.

[DHLP09] Bart De Schutter, W. P. M. H. Heemels, Jan Lunze, and Christophe Prieur. *Handbook of Hybrid Systems Control. Theory, Tools and Applications*, chapter 2 Survey of modelling, analysis, and control of hybrid systems, pages 33–55. Cambridge University Press, 2009.

[Dij75] Edsger W. Dijkstra. Guarded commands, nondeterminacy and formal derivation of programs. *Commun. ACM*, 18(8):453–457, August 1975.

[DLA06] Freddy Dumortier, Jaume Llibre, and Joan C. Artés. *Qualitative Theory of Planar Differential Systems*. Springer, 2006.

[DS95] Andreas Dolzmann and Thomas Sturm. Simplification of quantifier-free formulas over ordered fields. *Journal of Symbolic Computation*, 24:209–231, 1995.

[Eki03] Erdal Ekici. On the directional differentiability properties of the max-min function. *Boletín de la Asociación Matemática Venezolana*, X(1):35–42, 2003.

[ERNF15] Andreas Eggers, Nacim Ramdani, Nedialko S. Nedialkov, and Martin Fränzle. Improving the SAT modulo ODE approach to hybrid systems analysis by combining different enclosure methods. *Software and System Modeling*, 14(1):121–148, 2015.

[Fau99] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F4). *Journal of Pure and Applied Algebra*, 139(1–3):61 – 88, 1999.

[Fau02] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, ISSAC '02, pages 75–83, New York, NY, USA, 2002. ACM.

[FHT+07] Martin Fränzle, Christian Herde, Tino Teige, Stefan Ratschan, and Tobias Schubert. Efficient solving of large non-linear arithmetic constraint systems with complex boolean structure. *JSAT*, 1(3-4):209–236, 2007.

[Fil88] Alexei F. Filippov. *Differential Equations with Discontinuous Righthand Sides*. Mathematics and its Applications (Soviet Series). Kluwer Academic Publishers, 1988.

[FK04] Ansgar Fehnker and Bruce H Krogh. Hybrid system verification is not a sinecure. In *Automated Technology for Verification and Analysis*, pages 263–277. Springer, 2004.

[FLGD+11] Goran Frehse, Colas Le Guernic, Alexandre Donzé, Scott Cotton, Rajarshi Ray, Olivier Lebeltel, Rodolfo Ripado, Antoine Girard, Thao Dang, and Oded Maler. SpaceEx: Scalable Verification of Hybrid Systems. In Shaz Qadeer Ganesh Gopalakrishnan, editor, *Proc. 23rd International Conference on Computer Aided Verification (CAV)*, LNCS. Springer, 2011.

[FMQ+15] Nathan Fulton, Stefan Mitsch, Jan-David Quesel, Marcus Völp, and André Platzer. KeYmaera X: An axiomatic tactical theorem prover for hybrid systems. In Amy P. Felty and Aart Middeldorp, editors, *CADE*, LNCS. Springer, 2015.

[For91] Krister Forsman. *Constructive Commutative Algebra in Nonlinear Control Theory*. PhD thesis, Department of Electrical Engineering, Linköping University, S-581 83 Linköping, Sweden, 1991.

[Fre05] Goran Frehse. PHAVer: Algorithmic verification of hybrid systems past HyTech. In *HSCC*, pages 258–273. Springer, 2005.

[GAA+13] Georges Gonthier, Andrea Asperti, Jeremy Avigad, Yves Bertot, Cyril Cohen, François Garillot, Stéphane Le Roux, Assia Mahboubi, Russell O'Connor, Sidi Ould Biha, Ioana Pasca, Laurence

Rideau, Alexey Solovyev, Enrico Tassi, and Laurent Théry. A machine-checked proof of the odd order theorem. In Sandrine Blazy, Christine Paulin-Mohring, and David Pichardie, editors, *Interactive Theorem Proving*, volume 7998 of *Lecture Notes in Computer Science*, pages 163–179. Springer Berlin Heidelberg, 2013.

[GAC12] Sicun Gao, Jeremy Avigad, and Edmund M. Clarke. δ-complete decision procedures for satisfiability over the reals. In *Automated Reasoning - 6th International Joint Conference, IJCAR 2012, Manchester, UK, June 26-29, 2012. Proceedings*, pages 286–300, 2012.

[GGV10] Shuhong Gao, Yinhua Guan, and Frank Volny. A new incremental algorithm for computing Gröbner bases. In *Symbolic and Algebraic Computation, International Symposium, ISSAC 2010, Munich, Germany, July 25-28, 2010, Proceedings*, pages 13–19, 2010.

[Gin09] Jean-Marc Ginoux. *Differential Geometry Applied to Dynamical Systems*, volume 66 of *World Scientific Series on Nonlinear Science*. World Scientific, 2009.

[GKC13a] Sicun Gao, Soonho Kong, and Edmund M. Clarke. dreal: An SMT solver for nonlinear theories over the reals. In *Automated Deduction - CADE-24 - 24th International Conference on Automated Deduction, Lake Placid, NY, USA, June 9-14, 2013. Proceedings*, pages 208–214, 2013.

[GKC13b] Sicun Gao, Soonho Kong, and Edmund M. Clarke. Satisfiability modulo odes. In *Formal Methods in Computer-Aided Design, FMCAD 2013, Portland, OR, USA, October 20-23, 2013*, pages 105–112, 2013.

[Gon08] Georges Gonthier. The four colour theorem: Engineering of a formal proof. In *Computer Mathematics*, pages 333–333. Springer, 2008.

[Gor01] Alain Goriely. *Integrability and Nonintegrability of Dynamical Systems*. Advanced series in nonlinear dynamics. World Scientific, 2001.

[GP14a] Khalil Ghorbal and André Platzer. Characterizing algebraic invariants by differential radical invariants. In Erika Ábrahám and Klaus Havelund, editors, *TACAS*, volume 8413, pages 279–294. Springer, 2014.

[GP14b] Khalil Ghorbal and André Platzer. Characterizing algebraic invariants by differential radical invariants. Technical Report CMU-CS-13-129, Carnegie Mellon University, January 2014.

[GS97] Susanne Graf and Hassen Saïdi. Construction of abstract state graphs with PVS. In Orna Grumberg, editor, *Computer Aided Verification*, volume 1254 of *Lecture Notes in Computer Science*, pages 72–83. Springer Berlin Heidelberg, 1997.

[GSP14] Khalil Ghorbal, Andrew Sogokon, and André Platzer. Invariance of conjunctions of polynomial equalities for algebraic differential equations. In *Static Analysis - 21st International Symposium, SAS 2014, Munich, Germany, September 11-13, 2014. Proceedings*, pages 151–167, 2014.

[GSP15a] Khalil Ghorbal, Andrew Sogokon, and André Platzer. A hierarchy of proof rules for checking differential invariance of algebraic sets. In *Verification, Model Checking, and Abstract Interpretation - 16th International Conference, VMCAI 2015, Mumbai, India, January 12-14, 2015. Proceedings*, pages 431–448, 2015.

[GSP15b] Khalil Ghorbal, Andrew Sogokon, and André Platzer. A hierarchy of proof rules for checking positive invariance of algebraic and semi-algebraic sets. *Computer Languages, Systems & Structures*, 2015.

[GT08] Sumit Gulwani and Ashish Tiwari. Constraint-based approach for analysis of hybrid systems. In Aarti Gupta and Sharad Malik, editors, *Computer Aided Verification*, volume 5123 of *Lecture Notes in Computer Science*, pages 190–203. Springer Berlin Heidelberg, 2008.

[Gyf63] Elias P. Gyftopoulos. Lagrange stability by Liapunov's direct method. In *Proc. of Symposium on Reactor Kinetics and Control*, number TID-7662, pages 227–237, University of Arizona, 1963.

[GZ04] Hervé Guéguen and Janan Zaytoon. On the formal verification of hybrid systems. *Control Engineering Practice*, 12(10):1253 – 1267, 2004. Analysis and Design of Hybrid Systems.

[H́79] Otomar Hájek. Discontinuous differential equations, I. *Journal of Differential Equations*, 32(2):149 – 170, 1979.

[Har64] Philip Hartman. *Ordinary Differential Equations*. John Wiley & Sons, Inc., New York, 1964.

[Har72] Philip Hartman. On invariant sets and on a theorem of Ważewski. *Proceedings of the American Mathematical Society*, 32(2):pp. 511–520, 1972.

[HC08] Wassim M. Haddad and VijaySekhar Chellaboina. *Nonlinear Dynamical Systems and Control, a Lyapunov-based approach*. Princeton University Press, 2008.

[Hen96] Thomas A. Henzinger. The theory of hybrid automata. pages 278–292. IEEE Computer Society Press, 1996.

[HGH93] J.Y. Hung, W. Gao, and J.C. Hung. Variable structure control: a survey. *IEEE Transactions on Industrial Electronics*, 40(1):2–22, 1993.

[HHM+10] Thomas C. Hales, John Harrison, Sean McLaughlin, Tobias Nipkow, Steven Obua, and Roland Zumkeller. A revision of the proof of the Kepler Conjecture. *Discrete & Computational Geometry*, 44(1):1–34, 2010.

[HHWt97] Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-toi. HyTech: A Model Checker for Hybrid Systems. *Software Tools for Technology Transfer*, 1:460–463, 1997.

[HLLD09] W. P. M. H. Heemels, Daniel Lehmann, Jan Lunze, and Bart De Schutter. *Handbook of Hybrid Systems Control. Theory, Tools and Applications*, chapter 1 Introduction to hybrid systems, pages 3–30. Cambridge University Press, 2009.

[Hoa69] C. A. R. Hoare. An axiomatic basis for computer programming. *Commun. ACM*, 12(10):576–580, October 1969.

[HR08] Michael W Hofbaur and Theresa Rienmüller. Qualitative abstraction of piecewise affine systems. *Proc. QR*, 8:24–26, 2008.

[HW96] Ernst Hairer and Gerhard Wanner. *Solving Ordinary Differential Equations II (Stiff and Differential-Algebraic Problems)*. Springer Series in Computational Mathematics. Springer, second revised edition, 1996.

[Imm14] Fabian Immler. Formally verified computation of enclosures of solutions of ordinary differential equations. In *NASA Formal Methods - 6th International Symposium, NFM 2014, Houston, TX, USA, April 29 - May 1, 2014. Proceedings*, pages 113–127, 2014.

[Imm15] Fabian Immler. Verified reachability analysis of continuous systems. In Christel Baier and Cesare Tinelli, editors, *TACAS*, volume 9035 of *LNCS*. Springer, 2015.

[JdM12] Dejan Jovanović and Leonardo Mendonça de Moura. Solving non-linear arithmetic. In *Automated Reasoning - 6th International Joint Conference, IJCAR 2012, Manchester, UK, June 26-29, 2012. Proceedings*, pages 339–354, 2012.

[Jir98] Mats Jirstrand. *Constructive Methods for Inequality Constraints in Control*. PhD thesis, Department of Electrical Engineering, Linköping University, S-581 83 Linköping, Sweden, 1998.

[JSBP14] Paul B. Jackson, Andrew Sogokon, James P. Bridge, and Lawrence C. Paulson. Verifying hybrid systems involving transcendental functions. In *NASA Formal Methods - 6th International Symposium, NFM 2014, Houston, TX, USA, April 29 - May 1, 2014. Proceedings*, pages 188–202, 2014.

[KB00] Harry G. Kwatny and Gilmer L. Blankenship. *Nonlinear Control and Analytical Mechanics*. Birkhäuser Boston, 2000.

[KEH+09] Gerwin Klein, Kevin Elphinstone, Gernot Heiser, June Andronick, David Cock, Philip Derrin, Dhammika Elkaduwe, Kai Engelhardt, Rafal Kolanski, Michael Norrish, Thomas Sewell, Harvey Tuch, and Simon Winwood. seL4: formal verification of an OS kernel. In *Proceedings of the 22nd ACM Symposium on Operating Systems Principles 2009, SOSP 2009, Big Sky, Montana, USA, October 11-14, 2009*, pages 207–220, 2009.

[KGG+09] Stefan Kowalewski, Mauro Garavello, Hervé Guéguen, Gerlind Herberich, Rom Langerak, Benedetto Piccoli, Jan Willem Polderman, and Carsten Weise. *Handbook of Hybrid Systems Control. Theory, Tools and Applications*, chapter 3 Hybrid Automata, pages 59–85. Cambridge University Press, 2009.

[Kha92] Hassan K. Khalil. *Nonlinear Systems*. Macmillan Publishing Company, 1992.

[Kok95] Mieczysław M. Kokar. On consistent symbolic representations of general dynamic systems. *Systems, Man and Cybernetics, IEEE Transactions on*, 25(8):1231–1242, Aug 1995.

[Kui86] Benjamin Kuipers. Qualitative simulation. *Artificial intelligence*, 29(3):289–338, 1986.

[Lam77] Leslie Lamport. Proving the correctness of multiprocess programs. *IEEE Transactions on Software Engineering*, 3(2):125–143, March 1977.

[Ler09] Xavier Leroy. Formal verification of a realistic compiler. *Communications of the ACM*, 52(7):107–115, 2009.

[LG09] Colas Le Guernic and Antoine Girard. Reachability analysis of hybrid systems using support functions. In *Computer Aided Verification, 21st International Conference, CAV 2009, Grenoble, France, June 26 - July 2, 2009. Proceedings*, pages 540–554, 2009.

[Lie93] Sophus Lie. *Vorlesungen über continuierliche Gruppen mit Geometrischen und anderen Anwendungen*. Teubner, Leipzig, 1893.

[LJS+03] John Lygeros, Karl Henrik Johansson, Slobodan N. Simić, Jun Zhang, and Shankar S. Sastry. Dynamical properties of hybrid automata. *IEEE Trans. Automat. Contr.*, 48(1):2–17, 2003.

[LK93] Wood W. Lee and Benjamin Kuipers. A qualitative method to construct phase portraits. In *Proceedings of the 11th National Conference on Artificial Intelligence. Washington, DC, USA, July 11-15, 1993.*, pages 614–619, 1993.

[LLQ+10] Jiang Liu, Jidong Lv, Zhao Quan, Naijun Zhan, Hengjun Zhao, Chaochen Zhou, and Liang Zou. A calculus for hybrid csp. In Kazunori Ueda, editor, *Programming Languages and Systems*, volume 6461 of *Lecture Notes in Computer Science*, pages 1–15. Springer Berlin Heidelberg, 2010.

[LPY01] Gerardo Lafferriere, George J. Pappas, and Sergio Yovine. Symbolic reachability computation for families of linear vector fields. *J. Symb. Comput.*, 32(3):231–253, 2001.

[LS07] Youdong Lin and Mark A Stadtherr. Validated solutions of initial value problems for parametric ODEs. *Applied Numerical Mathematics*, 57(10):1145–1162, 2007.

[LW93] Rüdiger Loos and Volker Weispfenning. Applying linear quantifier elimination. *Comput. J.*, 36(5):450–462, 1993.

[Lya92] Alexander M. Lyapunov. *The general problem of stability of motion.* Kharkov Mathematical Society, Kharkov, 1892.

[LZZ11] Jiang Liu, Naijun Zhan, and Hengjun Zhao. Computing semi-algebraic invariants for polynomial dynamical systems. In *Proceedings of the ninth ACM international conference on Embedded software*, EMSOFT '11, pages 97–106, New York, NY, USA, 2011. ACM.

[LZZZ15] Jiang Liu, Naijun Zhan, Hengjun Zhao, and Liang Zou. Abstraction of elementary hybrid systems by variable transformation. In *FM 2015: Formal Methods - 20th International Symposium, Oslo, Norway, June 24-26, 2015, Proceedings*, pages 360–377, 2015.

[May89] Ernst W. Mayr. Membership in plynomial ideals over Q is exponential space complete. In *STACS 89, 6th Annual Symposium on Theoretical Aspects of Computer Science, Paderborn, FRG, February 16-18, 1989, Proceedings*, pages 400–406, 1989.

[May97] Ernst W. Mayr. Some complexity results for polynomial ideals. *J. Complexity*, 13(3):303–325, 1997.

[MB08] Oded Maler and Grégory Batt. Approximating continuous systems by timed automata. In Jasmin Fisher, editor, *Formal Methods in Systems Biology*, volume 5054 of *Lecture Notes in Computer Science*, pages 77–89. Springer Berlin Heidelberg, 2008.

[MBK10] Nader Motee, Bassam Bamieh, and Mustafa Khammash. Model reduction of polynomial dynamical systems using differential algebra. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 6195–6200. IEEE, 2010.

[Mis93] Bhubaneswar Mishra. *Algorithmic Algebra*. Texts and Monographs in Computer Science. Springer, 1993.

[MLK98] J. Strother Moore, Thomas W. Lynch, and Matt Kaufmann. A mechanically checked proof of the $AMD5_K86^{TM}$ floating point division program. *IEEE Trans. Computers*, 47(9):913–926, 1998.

[MM82] Ernst W Mayr and Albert R Meyer. The complexity of the word problems for commutative semigroups and polynomial ideals. *Advances in Mathematics*, 46(3):305 – 329, 1982.

[MMR11] Nadir Matringe, ArnaldoVieira Moura, and Rachid Rebiha. Generating invariants for non-linear hybrid systems by linear algebraic methods. In Radhia Cousot and Matthieu Martel, editors, *Static Analysis*, volume 6337 of *Lecture Notes in Computer Science*, pages 373–389. Springer Berlin Heidelberg, 2011.

[Nag42] Mitio Nagumo. Über die Lage der Integralkurven gewöhnlicher Differentialgleichungen. In *Proceedings of the Physico-Mathematical Society of Japan*, volume 24, pages 551–559, May 1942.

[Ned06] Nedialko S. Nedialkov. Interval Tools for ODEs and DAEs. In *12th GAMM - IMACS International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics (SCAN)*, pages 4–4, Sept 2006.

[NJC99] Nedialko S. Nedialkov, Kenneth R. Jackson, and George F. Corliss. Validated solutions of initial value problems for ordinary differential equations. *Applied Mathematics and Computation*, 105(1):21–68, 1999.

[NJN07] M. Neher, K. R. Jackson, and N. S. Nedialkov. On Taylor model based integration of ODEs. *SIAM Journal on Numerical Analysis*, 45(1):236–262, 2007.

[NLC11] Eva M. Navarro-López and Rebekah Carter. Hybrid automata: an insight into the discrete abstraction of discontinuous systems. *International Journal of Systems Science*, 42(11):1883–1898, 2011.

[NMKD91] Toyoaki Nishida, Kenji Mizutani, Atsushi Kubota, and Shuji Doshita. Automated phase portrait analysis by integrating qualitative and quantitative analysis. In *Proceedings of the 9th National Conference on Artificial Intelligence, Anaheim, CA, USA, July 14-19, 1991, Volume 2.*, pages 811–816, 1991.

[OL82] Susan Owicki and Leslie Lamport. Proving liveness properties of concurrent programs. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3):455–495, 1982.

[Olv98] Peter J Olver. *Applications of Lie groups to differential equations*, volume 107 of *Graduate Texts in Mathematics*. Springer, second edition, 1998.

[ORS92] Sam Owre, John M. Rushby, and Natarajan Shankar. PVS: A prototype verification system. In *Automated Deduction - CADE-11, 11th International Conference on Automated Deduction, Saratoga Springs, NY, USA, June 15-18, 1992, Proceedings*, pages 748–752, 1992.

[Par92] Patrick C Parks. A. M. Lyapunov's stability theory—100 years on. *IMA Journal of Mathematical Control and Information*, 9(4):275–303, 1992.

[Par00] Pablo A. Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. Engineering and applied science, control and dynamical systems, California Institute of Technology, May 2000.

[Pas11] Grant Olney Passmore. *Combined Decision Procedures for Nonlinear Arithmetics, Real and Complex*. PhD thesis, LFCS, University of Edinburgh, 2011.

[Pau98] Lawrence C. Paulson. The inductive approach to verifying cryptographic protocols. *Journal of Computer Security*, 6(1-2):85–128, 1998.

[PC08] André Platzer and Edmund M. Clarke. Computing differential invariants of hybrid systems as fixedpoints. In *Computer Aided Verification, 20th International Conference, CAV 2008, Princeton, NJ, USA, July 7-14, 2008, Proceedings*, pages 176–189, 2008.

[PdMJ10] Grant Olney Passmore, Leonardo Mendonça de Moura, and Paul B. Jackson. Gröbner basis construction algorithms based on theorem proving saturation loops. In *Decision Procedures in Software, Hardware and Bioware, 18.04. - 23.04.2010*, 2010.

[PJ04] Stephen Prajna and Ali Jadbabaie. Safety verification of hybrid systems using barrier certificates. In Rajeev Alur and GeorgeJ. Pappas, editors, *Hybrid Systems: Computation and Control*, volume 2993 of *Lecture Notes in Computer Science*, pages 477–492. Springer Berlin Heidelberg, 2004.

[PJP07] S. Prajna, A. Jadbabaie, and G.J. Pappas. A framework for worst-case and stochastic safety verification using barrier certificates. *Automatic Control, IEEE Transactions on*, 52(8):1415–1428, 2007.

[Pla08] André Platzer. Differential dynamic logic for hybrid systems. *J. Autom. Reas.*, 41(2):143–189, 2008.

[Pla10a] André Platzer. Differential-algebraic dynamic logic for differential-algebraic programs. *J. Log. Comput.*, 20(1):309–352, 2010.

[Pla10b] André Platzer. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Springer, Heidelberg, 2010.

[Pla11] André Platzer. Logic and compositional verification of hybrid systems (invited tutorial). In Ganesh Gopalakrishnan and Shaz Qadeer, editors, *CAV*, volume 6806 of *LNCS*, pages 28–43. Springer, 2011.

[Pla12a] André Platzer. A differential operator approach to equational differential invariants. In Lennart Beringer and Amy Felty, editors, *ITP*, volume 7406 of *LNCS*, pages 28–48. Springer, 2012.

[Pla12b] André Platzer. The structure of differential invariants and differential cut elimination. *Logical Methods in Computer Science*, 8(4):1–38, 2012.

[Pla15] André Platzer. A uniform substitution calculus for differential dynamic logic. In Amy P. Felty and Aart Middeldorp, editors, *CADE*, LNCS. Springer, 2015.

[Poi81] Henri Poincaré. Mémoire sur les courbes définies par une équation différentielle, 1ère partie (in French). *Journal de mathématiques pures et appliquées*, 7:375–422, 1881.

[Poi82] Henri Poincaré. Mémoire sur les courbes définies par une équation différentielle, 2nde partie (in French). *Journal de mathématiques pures et appliquées*, 3:251–296, 1882.

[Poi85] Henri Poincaré. Sur les courbes définies par les équations différentielles, 3ème partie (in French). *Journal de mathématiques pures et appliquées*, 4:167–244, 1885.

[Pow59] John E. Powers. Elimination of special functions from differential equations. *Commun. ACM*, 2(3):3–4, March 1959.

[PP05] Antonis Papachristodoulou and Stephen Prajna. Analysis of non-polynomial systems using the sum of squares decomposition. In Didier Henrion and Andrea Garulli, editors, *Positive Polynomials in Control*, volume 312 of *Lecture Notes in Control and Information Science*, pages 23–43. Springer Berlin Heidelberg, 2005.

[PQ08] André Platzer and Jan-David Quesel. KeYmaera: A hybrid theorem prover for hybrid systems. In Alessandro Armando, Peter

Baumgartner, and Gilles Dowek, editors, *IJCAR*, volume 5195 of *LNCS*, pages 171–178. Springer, 2008.

[PR05] Stephen Prajna and Anders Rantzer. Primal–dual tests for safety and reachability. In Manfred Morari and Lothar Thiele, editors, *Hybrid Systems: Computation and Control*, volume 3414 of *Lecture Notes in Computer Science*, pages 542–556. Springer Berlin Heidelberg, 2005.

[Pra95] Vaughan Pratt. Anatomy of the Pentium bug. In PeterD. Mosses, Mogens Nielsen, and MichaelI. Schwartzbach, editors, *TAPSOFT '95: Theory and Practice of Software Development*, volume 915 of *Lecture Notes in Computer Science*, pages 97–107. Springer Berlin Heidelberg, 1995.

[Pra05] Stephen Prajna. *Optimization-Based Methods for Nonlinear and Hybrid Systems Verification*. PhD thesis, Engineering and Applied Science, California Institute of Technology, Pasadena, California, 2005.

[QML+15] Jan-David Quesel, Stefan Mitsch, Sarah Loos, Nikos Aréchiga, and André Platzer. How to model and prove hybrid systems with KeYmaera: A tutorial on safety. 2015.

[Rat06] Stefan Ratschan. Efficient solving of quantified inequality constraints over the real numbers. *ACM Transactions on Computational Logic*, 2006.

[Red72] R. M. Redheffer. The theorems of Bony and Brezis on flow-invariant sets. *The American Mathematical Monthly*, 79(7):pp. 740–747, 1972.

[Ric68] Daniel Richardson. Some undecidable problems involving elementary functions of a real variable. *Journal of Symbolic Logic*, 33(4):514–520, 12 1968.

[RS06] Stefan Ratschan and Zhikun She. Constraints for continuous reachability in the verification of hybrid systems. In *Artificial Intelligence and Symbolic Computation, 8th International Conference, AISC 2006, Beijing, China, September 20-22, 2006, Proceedings*, pages 196–210, 2006.

[RS07] Stefan Ratschan and Zhikun She. Safety verification of hybrid systems by constraint propagation-based abstraction refinement. *ACM Transactions on Embedded Computing Systems*, 6(1), February 2007.

[RS10] Stefan Ratschan and Zhikun She. Providing a basin of attraction to a target region of polynomial systems by computation of Lyapunov-like functions. *SIAM J. Control and Optimization*, 48(7):4377–4394, July 2010.

[Sac90a] Elisha Sacks. Automatic qualitative analysis of dynamic systems using piecewise linear approximations. *Artificial Intelligence*, 41(3):313–364, 1990.

[Sac90b] Elisha Sacks. A dynamic systems perspective on qualitative simulation. *Artificial Intelligence*, 42(2–3):349–362, 1990.

[San10] Sriram Sankaranarayanan. Automatic invariant generation for hybrid systems using ideal fixed points. In *Proceedings of the 13th ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '10, pages 221–230, New York, NY, USA, 2010. ACM.

[SAZ14] Wen Su, Jean-Raymond Abrial, and Huibiao Zhu. Formalizing hybrid systems with Event-B and the Rodin Platform. *Science of Computer Programming*, 94:164–202, 2014.

[Sch93] Dana Schlomiuk. Algebraic and geometric aspects of the theory of polynomial vector fields. In Dana Schlomiuk, editor, *Bifurcations and Periodic Orbits of Vector Fields*, volume 408 of *NATO ASI Series*, pages 429–467. Springer Netherlands, 1993.

[SDI08] Sriram Sankaranarayanan, Thao Dang, and Franjo Ivančić. Symbolic model checking of hybrid systems using template polyhedra. In *Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2008, Budapest, Hungary, March 29-April 6, 2008. Proceedings*, pages 188–202, 2008.

[SEK99] Olaf Stursberg, Sebastian Engell, and Stefan Kowalewski. Timed approximations of hybrid processes for controller verification. In *Proceedings of the 14th IFAC World Congress*, pages 73–78, Beijing, China, 1999.

[SGJP16] Andrew Sogokon, Khalil Ghorbal, Paul B. Jackson, and André Platzer. A method for invariant generation for polynomial continuous systems. In Barbara Jobstmann and K. Rustan M. Leino, editors, *Verification, Model Checking, and Abstract Interpretation - 17th International Conference, VMCAI 2016, St. Petersburg, Florida, USA, January 17-19, 2016. Proceedings*, volume 9583 of *LNCS*. Springer, 2016.

[SGT08] Ricardo G. Sanfelice, Rafal Goebel, and Andrew R. Teel. Generalized solutions to hybrid dynamical systems. *ESAIM: Control, Optimisation and Calculus of Variations*, 14:699–724, 10 2008.

[SJ15] Andrew Sogokon and Paul B. Jackson. Direct formal verification of liveness properties in continuous and hybrid dynamical systems. In

FM 2015: Formal Methods - 20th International Symposium, Oslo, Norway, June 24-26, 2015, Proceedings, pages 514–531, 2015.

[SK03] Olaf Stursberg and Bruce H. Krogh. Efficient representation and computation of reachable sets for hybrid systems. In *Hybrid Systems: Computation and Control, 6th International Workshop, HSCC 2003 Prague, Czech Republic, April 3-5, 2003, Proceedings*, pages 482–497, 2003.

[SKA01] James A Stiver, Xenofon D Koutsoukos, and Panos J Antsaklis. An invariant-based approach to the design of hybrid control systems. *International Journal of Robust and Nonlinear Control*, 11(5):453–478, 2001.

[SKHP96] Olaf Stursberg, Stefan Kowalewski, Ingo Hoffmann, and Jörg Preußig. Comparing timed and hybrid automata as approximations of continuous systems. In *Hybrid Systems IV*, pages 361–377, 1996.

[SP95] A. M. Samoilenko and N. A. Perestyuk. *Impulsive Differential Equations*, volume 14 of *World Scientific Series on Nonlinear Science*. World Scientific, 1995.

[SPW12] Christoffer Sloth, George J. Pappas, and Rafael Wiśniewski. Compositional safety analysis using barrier certificates. In *Proceedings of the 15th ACM international conference on Hybrid Systems: Computation and Control*, HSCC '12, pages 15–24, New York, NY, USA, 2012. ACM.

[SSM08] Sriram Sankaranarayanan, Henny B. Sipma, and Zohar Manna. Constructing invariants for hybrid systems. *Formal Methods in System Design*, 32(1):25–55, 2008.

[Str94] Steven H. Strogatz. *Nonlinear Dynamics And Chaos*. Westview Press, 1994.

[SV87] Michael A. Savageau and Eberhard O. Voit. Recasting nonlinear differential equations as S-systems: a canonical nonlinear form . *Mathematical Biosciences*, 87(1):83 – 115, 1987.

[SW11] Christoffer Sloth and Rafael Wiśniewski. Verification of continuous dynamical systems by timed automata. *Formal Methods in System Design*, 39(1):47–82, 2011.

[SW13] Christoffer Sloth and Rafael Wiśniewski. Complete abstractions of dynamical systems by timed automata. *Nonlinear Analysis: Hybrid Systems*, 7(1):80 – 100, 2013. IFAC World Congress 2011.

[Tar51] Alfred Tarski. A decision method for elementary algebra and geometry. *Bulletin of the American Mathematical Society*, 59, 1951.

[Tiw08a] Ashish Tiwari. Abstractions for hybrid systems. *Formal Methods in System Design*, 32(1):57–83, 2008.

[Tiw08b] Ashish Tiwari. Generating box invariants. In Magnus Egerstedt and Bud Mishra, editors, *Hybrid Systems: Computation and Control*, volume 4981 of *Lecture Notes in Computer Science*, pages 658–661. Springer Berlin Heidelberg, 2008.

[TK02] A. Tiwari and G. Khanna. Series of abstractions for hybrid automata. In C. J. Tomlin and M. R. Greenstreet, editors, *Hybrid Systems: Computation and Control HSCC*, volume 2289 of *LNCS*, pages 465–478. Springer, March 2002.

[TK04] A. Tiwari and G. Khanna. Nonlinear systems: Approximating reach sets. In R. Alur and G. Pappas, editors, *Hybrid Systems: Computation and Control HSCC*, volume 2993 of *LNCS*, pages 600–614. Springer, March 2004.

[TT09] Ankur Taly and Ashish Tiwari. Deductive verification of continuous dynamical systems. In Ravi Kannan and K. Narayan Kumar, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 4 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 383–394, 2009.

[UGS99] Vadim Utkin, Jürgen Guldner, and Jingxin Shi. *Sliding Mode Control in Electromechanical Systems*. Series in Systems and Control. Taylor & Francis, 1999.

[Utk92] Vadim Utkin. *Sliding Modes in Control and Optimization*. Communications and Control Engineering Series. Springer-Verlag, 1992.

[Wal98] Wolfgang Walter. *Ordinary Differential Equations*. Graduate Texts in Mathematics. Springer New York, 1998.

[Wei97] Volker Weispfenning. Quantifier elimination for real algebra - the quadratic case and beyond. *Appl. Algebra Eng. Commun. Comput.*, 8(2):85–101, 1997.

[Wig03] Stephen Wiggins. *Introduction to Applied Nonlinear Dynamical Systems and Chaos*. Texts in Applied Mathematics. Springer, second edition, 2003.

[WLW13] Ta-Chung Wang, Sanjay Lall, and Matthew West. Polynomial level-set method for polynomial system reachable set estimation. *IEEE Transactions on Automatic Control*, 58(10):2508–2521, 2013.

[Wu10] Zili Wu. Tangent cone and contingent cone to the intersection of two closed sets. *Nonlinear Analysis: Theory, Methods & Applications* , 73(5):1203 – 1220, 2010.

[Yor67] James A. Yorke. Invariance for ordinary differential equations. *Mathematical Systems Theory*, 1(4):353–372, 1967.

[Yor68] James A. Yorke. Correction: Invariance for ordinary differential equations. *Mathematical Systems Theory*, 2(4):381, 1968.

[Yun76] David Y. Y. Yun. On square-free decomposition algorithms. In *Proceedings of the Third ACM Symposium on Symbolic and Algebraic Computation*, SYMSAC '76, pages 26–35, New York, NY, USA, 1976. ACM.

[Zha94] Feng Zhao. Extracting and representing qualitative behaviors of complex systems in phase space. *Artif. Intell.*, 69(1-2):51–92, 1994.

[ZTB06] Mohamed H. Zaki, Sofiène Tahar, and Guy Bois. Abstraction based verification of analog circuits using computer algebra and constraint solving. In *Proc. International Workshop on Symbolic Methods and Applications to Circuit Design*, 2006.

[ZTB07] Mohamed H. Zaki, Sofiène Tahar, and Guy Bois. A symbolic approach for the safety verification of continuous systems. In *Proceedings of the International Conference on Computational Sciences*, pages 93–100, 2007.

[ZW12] Eva Zerz and Sebastian Walcher. Controlled invariant hypersurfaces of polynomial control systems. *Qualitative Theory of Dynamical Systems*, 11(1):145–158, 2012.

[ZWG10] Eva Zerz, Sebastian Walcher, and Fadime Güçlü. Controlled invariant varieties of polynomial control systems. In *Proceedings of the 19th International Symposium on Mathematical Theory of Networks and Systems*, 2010.

[ZYZ+14] Hengjun Zhao, Mengfei Yang, Naijun Zhan, Bin Gu, Liang Zou, and Yao Chen. Formal verification of a descent guidance control program of a lunar lander. In *FM*, pages 733–748, 2014.

[ZZK13] Hengjun Zhao, Naijun Zhan, and Deepak Kapur. Synthesizing switching controllers for hybrid systems by generating invariants. In *Theories of Programming and Formal Methods*, pages 354–373, 2013.